“十二五”职业教育国家规划教材
经全国职业教育教材审定委员会审定

21 世纪高等职业教育计算机系列规划教材

基于 ASP.NET 的 Web 应用开发技术实用教程

（第 2 版）

方玉燕　主　编

蓝建平　副主编

電子工業出版社

Publishing House of Electronics Industry

北京 • BEIJING

内 容 简 介

本书采用基于C#的ASP.NET 4.5的技术，用项目教学法将ASP.NET的知识点融入教学案例当中。项目的选择遵循易于理解、简单而又全面的原则，使学生随着教材内容的推进在不知不觉中掌握Web开发技术的精髓。书中遵循“理论来源于实践，又指导实践”的思想，采用“实践演练→知识点学习→任务拓展实战”的方式来编排教学内容，弥补了大多数教材中理论教学与实践脱节的不足。

本书采用项目化教学方式，共分为6个项目。项目可以归为两类，其中有3个项目是独立的功能模块，融合了Web开发所需的一些常用技术，这3个功能模块分别是我的第一个网站、信息处理和Ajax聊天室；另外4个项目以3个完整的网站为纽带，把项目分解为多个子任务的方式来完成项目的开发，同时把ASP.NET 4.5的基本知识体系融入其中，其中项目2和项目3采用同一个网站，项目3可以说是对项目2进行了完善，这3个完整的网站是网络通讯录、企业网站和聊天室。

本书不仅适合ASP.NET初学者学习，也可供广大Web程序开发人员参考。

图书在版编目（CIP）数据

基于ASP.NET的Web应用开发技术实用教程 / 方玉燕主编. —2版. —北京：电子工业出版社，2015.1
（“十二五”职业教育国家规划教材）

ISBN 978-7-121-24131-4

Ⅰ. ①基… Ⅱ. ①方… Ⅲ. ①网页制作工具—程序设计—高等职业教育—教材 Ⅳ. ①TP393.092

中国版本图书馆CIP数据核字（2014）第191750号

策划编辑：徐建军（xujj@phei.com.cn）
责任编辑：郝黎明
印　　刷：
装　　订：
出版发行：电子工业出版社
　　北京市海淀区万寿路173信箱　邮编　100036
开　　本：787×1 092　1/16　印张：17.75　字数：454.4千字
版　　次：2010年12月第1版
　　2015年1月第2版
印　　次：2015年1月第1次印刷
印　　数：3 000册　定价：36.00元

凡所购买电子工业出版社图书有缺损问题，请向购买书店调换。若书店售缺，请与本社发行部联系，联系及邮购电话：（010）88254888。

质量投诉请发邮件至zlts@phei.com.cn，盗版侵权举报请发邮件至dbqq@phei.com.cn。

服务热线：（010）88258888。

前　言

近年来，软件产业以十分惊人的速度发展，软件和信息服务业将成为世界第一大产业。

在人才方面，我国软件业人才的供需缺口越来越大，高级人才不仅奇缺，同时从事软件产业基础性工作的软件蓝领也奇缺，出现了前所未有的“人才饥荒”。各地对软件人才的争夺战也不断升温，每年几十万的计算机专业人员毕业待业，显得高不成，低不就，根源在哪里？究其原因是：理论与实践的脱节。一般学生毕业以后到公司要经过 6～12 个月的培训才能胜任工作。因此培训新人需要花费大量的时间和费用，让企业付着工资对学生再进行一次补习式教育，造成了教育资源的浪费。

互联网的升温带动了企业对 Web 开发人才的需求量，近几年各类学校都非常重视 Web 开发技术人才的培养，同时各级政府也非常重视这方面人才的培养。以浙江省为例，为培养高校 Web 开发的人才，开设了多种比赛，如浙江省大学科技竞赛委员会在电子商务竞赛和多媒体技术竞赛中都开设了 Web 网站技术的比赛项目，同时行业协会也开设了大量这方面的比赛，其目的就是为了加强 Web 开发技术方面人才的培养，以赛促教，以赛促学。ASP.NET 技术因其入门比较容易，开发周期短，能满足众多中小企业的需求，也更适合高职学生的学习要求。

作者从事 Web 网站开发课程教学，尝试过各种不同的教学模式。同时也常为找一本符合自己教学理念的教材而苦恼。经过一段时间的探索，在采用了完整案例与理论教学体系相融合的方式后，发现学生的学习成绩得到了大幅度的提高，企业更喜欢这类学生。

目前同类教材虽然很多。但在案例的编排中，有的过分重视组成 Web 开发中所涉及的功能模块，“只见树木不见森林”；有的却安排了完整的项目案例，但又缺少理论教学。“理论来源于实践，又指导实践”，实践尽管很重要，但没有了理论的指导，学生的整个学习体系也会像水中浮萍。

在教学过程中，如何“教”固然重要，笔者认为如何“学”更加重要。一位好教师不仅体现在能教会学生多少知识，而更重要的是要教会学生如何去学习。本书在编写过程中融入了这个思路。

为激发学生的学习兴趣，在项目的选择上尽量选择与学生生活、就业等密切相关的项目。全书分为 6 个项目，每个项目根据其功能模块分解为几个子任务。各任务的实践演练环节把相关的知识体系以归纳、总结、深入的方式引领出来，起到“抛砖引玉”的作用。经过理论阶段的学习，接着安排了项目拓展部分，这部分有的是与实践演练相类似的功能模块，有的是知识体系的深入练习，很好地融入了“理论来源于实践，又指导实践”的理念。

项目 1 我的第一个网站，用一个极简单的案例开始课程学习。通过引领学生上网，了解什么是 Web 网站及与网站的相关知识。在此项目中使学生接触开发网站所需的基本知识及认识 ASP.NET 的结构体系。

项目 2 是一个通讯录网站，通过对通讯录的分析，设计网络通讯录的功能模块。在完成项目的注册功能后，学习 ASP.NET 的 Web 控件和验证控件及其相关的知识；通过登录，学习如何设计自己的用户控件及代码重用；通过各页面间的调用，学习 ASP.NET 三大内置对象；为实现对联系人的管理，掌握用 ADO.NET 连接方式访问数据库。书中的实战演练与拓展训练两

部分内容完成了整个网站的开发。

项目 3 信息验证，许多网站面临着信息验证的问题，能让学生了解目前网站中常用的技术验证码验证方式，也能初步了解威胁网站的一些因素，本项目在项目 2 的基本上，通过图文验证、邮件验证了解网站的一些防备技术，也学习了图文技术和邮件技术在 ASP.NET 中的应用。

项目 4 信息处理，本项目主要介绍如何对网站中最重要信息（文字、图片和文件）的处理。由于 ASP.NET 所提供的控件在处理这些信息上有很大的局限，所以在此项目中将介绍第三方控件的应用。

项目 5 企业网站，此项目初步介绍了多层架构项目开发的概念、ASP.NET 的母版页与皮肤技术、ADO.NET 非连接方式访问数据库及常用的数据绑定控件和导航控件等。通过对这个项目的学习，使学生掌握目前 Web 开发中常见的事企业网站的功能结构、网站的安全管理与配置。学完本项目，还可以安排学生参加一些实际的企业网站开发工作，这样更容易激发学生的学习兴趣。

项目 6Ajax 聊天室也是网站的常用功能，在这个项目中介绍 ASP.NET 4.5 中新集成的 Ajax 技术及控件。Ajax 技术在 ASP.NET 2.0 中就已经出现，但要通过安装相关组件才能使用。

书中尽量体现“成果引领、兴趣驱动、项目导向、团队合作”的教学理论。在教学中建议本课程以学习小组的方式展开学习，所以虽然前 6 个项目不要求以团队的方式完成，但却有团队合作考核要求。Web 项目开发涉及领域广泛，.NET 平台技术可以说是博大精深。俗话说，“术业有专攻”，每个学生都有自己的专长，采用小组学习方式不仅可以在学习上互相帮助、互相鼓励，还可以实现对知识体系的弥补。

通过对本书的学习，学生可以轻松实现企业宣传网站的开发，能胜任参加大型网站开发团队的开发工作。

本书由嘉兴职业技术学院的方玉燕老师组织编写并担任主编。项目 1～项目 4 由方玉燕老师编写，项目 5 和项目 6 由蓝建平老师编写；李玉清教授为本书的编写结构提出了大量的建设性意见；梅飞龙和蒋睿参与书稿的部分项目编写与查错工作；嘉兴中易软件公司的技术部经理钱明华先生、上海鑫思形象策划有限公司经理李勤峰先生、嘉兴微软技术中心技术总监王利华先生在修订过程中提出了大量宝贵意见，在此一并表示感谢。

为了方便教师教学，本书配有电子教学课件及程序源代码、软件开发各阶段的文档模板和相关资料，请有此需要的教师登录华信教育资源网（www.hxedu.com.cn）注册后免费进行下载，或到 www.jxwebjpkc.net 网站下载，如有问题可在网站留言板留言或与电子工业出版社联系（E-mail:hxedu@phei.com.cn），也可以与作者联系（E-mail：83981703@qq.com）。

由于项目式教学法正处于经验积累和改进过程中，所以虽然编写本书花了较长时间，并经过多次改稿，但书中难免存在疏漏和不足，希望同行专家和读者能给予批评和指正。

编　者

目　录

项目一

我的第一个网站

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 掌握网站设计的基本概念；
- ✧ 熟悉网站建设的流程；
- ✧ 了解如何在站点结构、站点风格等方面规划网站；初步了解网站架构模式；
- ✧ 掌握创建站点的基本步骤及站点的管理；
- ✧ 理解 ASP.NET 常用的网站集成开发环境 Visual Studio 2012；
- ✧ 掌握安装和配置 ASP.NET 开发环境；
- ✧ 掌握 ASP.NET 的页面结构、组成元素；
- ✧ 掌握 ASP.NET 的代码格式。

技能目标

通过对本项目的学习，应该具备下面的能力：

- ✧ 能安装并使用 Visual Studio 2012 集成开发环境；
- ✧ 能对 ASP.NET 开发环境进行配置；
- ✧ 能创建生成并浏览 ASP.NET Web 网站项目；
- ✧ 能创建 Web 窗体页面；
- ✧ 能发布网站。

教学建议

本项目计划总学时为：8 学时。

- ✧ 情境介绍：0.5 学时；
- ✧ 任务 1：1.5 学时；
- ✧ 任务 2：2 学时；
- ✧ 任务 3：4 学时。

在本项目开始授课时，教师可以有目的地带领学生在网站进行冲浪，了解各类网站的相同点与不同点，激发学生的学习积极性，初步建立学生今后学习的目标。

在本项目的教学中有两个点可以采用讨论的方式，第一个是进行网站冲浪后学生进行讨论总结网站的特点和目前网站的流行功能，第二个是一个好的网站应该具备哪些功能。

1.1 情境介绍

随着 Internet 技术的飞速发展，越来越多的人喜欢在网上搜索自己需要的资料。网站已逐渐成为政府、企业和个人对外展示、信息沟通最方便快捷的桥梁，同时网站也成为宣传、产品资讯发布、展现个性的营销舞台。

在开始学习之前，先来看一个如图 1-1 所示的网站，这个网站非常简单，打开网站后，在文本框内输入您的名字单击“确认”按钮后，会跳出如图 1-2 所示的网页内容，这个网页会根据您输入姓名的不同而显示不同的内容。

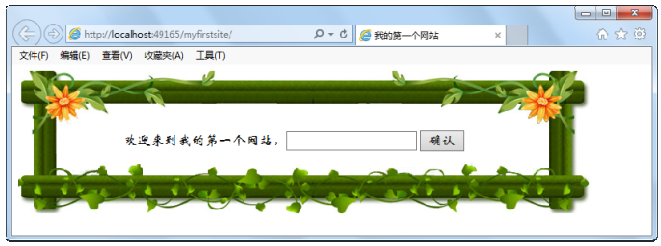

图 1-1 简单网站首页

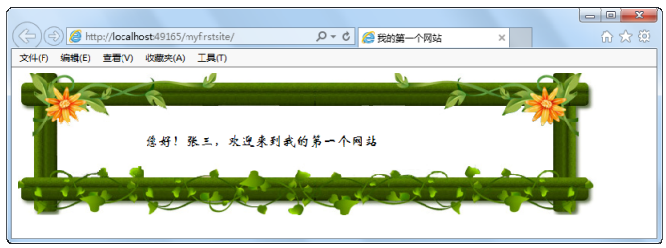

图 1-2 输入姓名后显示的页面

1.2 任务 1 网站建设的基本知识

任何一个项目的开发都是一个系统的工程，对于一个 Web 应用程序的开发者来说，不管是做一个只有几页的简单的网站，还是做一个复杂的包含各类新技术的网站，都要在网站建设前做好基本的准备工作。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）了解网站的基本结构；

（2）了解网站的组成要素；

（3）了解如何建设一个好的网站。

1.2.1 认识网站

在正式开始网站建设之前，首先要对网站有一个初步的认识。下面通过对几个网站的访问

来了解网站的基本内容。

一、实战演练

1. 网易网站

通过在浏览器中输入 www.163.com，打开网易网站。从如图 1-3 所示网易首页的内容可以看到，网页上充满了各类信息及信息的导航，这些信息服务是免费的，但在页面上充满了许多广告是收费的，是这类门户网站的主要收入之一。这类网站的主要特点是信息量大、广告多，充分利用页面的空间，文字小、密度大，而且主页的长度有多个屏幕。这类网站主要利用信息的快速更新来吸引网民，目前我国像这类较大的信息资源服务网站还有搜狐（www.sohu.com）、新浪（www.sina.com）等，是互联网用户生活中不可或缺的部分。

图 1-3 网易首页

2. 政府网站

如图 1-4 所示的政府网站的设计采用了传统的设计方法，在整个网站的布局中严谨而庄重，整个页面的色彩统一、主题突出。在这类网站的设计过程中充分考虑了使用不同显示器分辨率的用户，页面的大小正好充满一个屏幕，用户可以一目了然地浏览主页的全部内容。

3. 当当网网上购物中心

如图 1-5 所示的当当网是一个典型的电子商务网站。这类网站的设计模式与网易这类信息门户网站有相似之处，整个网站充满着产品信息和产品广告，信息量大。随着电子商务这类商业运作模式在网民中的认可，这类网站在近几年得到了快速发展。

图 1-4　某政府网站首页

图 1-5　当当网网站首页

二、知识点

1. 网站类型

目前 Internet 上的网站很多，对网站的分类也有很多种，大致可以分为信息提供型、电

子商务型和政府企业门户型。不同网站可能属于不同的类型，即使相同类型的网站也可能在内容、服务和风格上千变万化、各不相同，但作为一个完整的功能实体，它们又具有很多相同的特征。

目前在互联网中常见的网站还有：

1）导航网站

这类网站包括搜索引擎和网站目录。用户通过输入所需查找信息的几个关键字就可以快速查询到所需的网站或内容，如百度（www.baidu.com）。导航网站的另一种形式是对互联网上的网站设立分类导航，用户通过单击上面的超链接就可以转到浏览的目标网站，如网址之家（www.hao123.com）。

2）教育网站

这类网站极大地方便了人们继续学习和提升知识。这类网站除了一些专门的远程教育网站外，还包括一些科普网站、个人学习网站、行业学习网站等。如 CSDN 社区中心就是计算机技术类大型学习社区，为 IT 专业技术人员提供了最全面的信息传播和学习服务的平台，学习者不仅可以在上面找到自己想要的学习资料，也可以通过这个网站解决自己的技术难题。

3）财经网站

这类网站可提供强大的咨询、交流平台，协助客户迅速建立自己的客户群体，同时将各种应用系统、数据资源和互联网资源集成到一个信息管理平台上，并以统一的用户界面提供给用户。

4）娱乐网站

这类网站主要提供各种娱乐方式，如在线游戏、在线影院等都是典型的娱乐网站。

2. 网页基本元素

网页的基本元素包括标题、网站 LOGO、页眉、页脚、主体内容、功能区、导航区、广告栏等。这些元素在网页的位置安排，就是网页的整体布局。

1）标题

每个网页的最顶端都有一条信息，它往往出现在浏览器的标题栏，而非网页中，但是它也是网页布局中的一部分。这条信息是对这个网页中主要内容的提示，即标题。

2）网站 LOGO

LOGO 是网站所有者对外宣传自身形象的工具。LOGO 集中体现了这个网站的文化内涵和内容定位，使人们在看到 LOGO 标志的同时，自然地产生联想，从而对所代表事物产生认同。LOGO 是网站形象的重要体现。对于精美的网站，LOGO 更是它的灵魂所在。LOGO 也是与其他网站链接及让其他网站链接的标志和门户。一个好的 LOGO 应具备以下条件：符合国际标准、精美、独特，与网站的整体风格相融，能够体现网站的类型、内容和风格。

3）页眉

网页的上端即页面的页眉。并不是在所有的网页中都有页眉，一些特殊的网页就没有明确划分出页眉。页眉往往在一个页面中占据相当重要的位置，容易引起浏览者的注意，所以很多网站都会在页眉中设置宣传本网站的内容，如网站宗旨、网站 LOGO 等，也有一些网站将这个“黄金地段”作为广告位出租。

4）页脚

网页的最底端部分被称为页脚，页脚部分通常被用来介绍网站所有者的具体信息和联络方式，如名称、地址、联系方式、版权信息等。其中一些内容被做成标题式的超链接，引导浏览者进一步了解详细的内容。

5）功能区

功能区是网站主要功能的集中表现，一般位于网页的右上方或右侧边栏。功能区包括：电子邮件、信息发布、用户名注册、登录网站等内容。有些网站使用 IP 定位功能，定位浏览者所在地，然后可在功能区显示当地的天气、新闻等个性化信息。

6）主体内容

主体内容一般由图片和文档构成，现在一些网站的主体内容中还加入了视频、音频等多媒体元素。由于人们的阅读习惯是由上至下、由左至右，所以主体内容的分布也是这个规律，依照重要到不重要的顺序安排内容。在主体内容中，左上方的内容是最重要的。

主体内容是网页中最重要的元素。主体内容并不完整，往往由下一级内容的标题、内容提要、内容摘编的超链接构成。主体内容借助超链接，可以利用一个页面高度概括几个页面所表达的内容，而首页的主体内容甚至能在一个页面中高度概括整个网站的内容。

7）导航区

如果说主体内容重要的话，那么导航区的重要性与其不相上下，甚至导航区的设计可以成为一种独立的设计，与网页布局设计分庭抗礼。之所以说导航区重要，是因为其所在位置左右着整个网页布局的设计。导航区一般分为 4 种位置，分别是左侧、右侧、顶部和底部。一般网站使用的导航区都是单一的，但是也有一些网站为了使网页更便于浏览者操作，增加可访问性，往往采用了多导航技术，如 Yahoo!网站采用了左侧导航与底部导航相结合的方式。但是无论采用几个导航区，网站中每个页面的导航区位置均是固定的。

8）广告区

广告区是网站实现盈利或自我展示的区域，一般位于网页的页眉、右侧和底部。广告区内容以文字、图像、Flash 动画为主，通过吸引浏览者单击链接的方式达到广告效果。广告区设置要明显、合理、引人注目，这对整个网站的布局很重要。

3. 网页的整体布局结构

网页的基本元素包括标题在网页中的位置安排，就是网页的整体布局。

网页布局都是有一定规则的，纵观各类网页可以归纳为：左右对称结构布局、“同”字型结构布局、“回”字型结构布局、“匡”字型结构布局、“丁”字型结构布局、自由式结构布局、“另类”结构布局等。

4. 网站建设的基本原则

建设网站是一件很容易的事，但建设一个好网站是一件困难的事。网站建设的过程中，对网站规模、业务背景要进行深入分析，对结构规划、页面设计及数据库设计等每一步都要做完整的策划。事前全面、充分的准备，将有效地提高后续工作的效率和质量，如果网站的设计人员在网站项目开始时没有做好充分的准备，那么在网站的设计、维护过程中就容易出现各种问题。

1）网站的总体目标

网站建设的第一步就是确定整体目标，很多网站因为缺少清晰的目标而失败。网站项目能

否最终获得成功，取决于能否对各种相关需求进行有效的收集和整理，需求来自于潜在访问者及网站的所有者。如何更好地了解、分析、明确用户需求，保证网站项目的成功，是每个网站项目管理者需要面对的问题。在确定目标时一定要考虑的问题是：这个网站能够做什么？为什么要做这个网站？

在实际工作中，设计人员会发现，不断有各种需求出现。这些需求来自于设计人员、客户的建设性意见或者灵感，设计人员应该将其中积极可行的需求转化为相应的功能。相反，一些不良的需求会影响项目的稳定性，这样的需求包括客户不断否定、修改前期提出的需求，在预算不充分的情况下实现某些高成本的需求等。

实际上，一个网站不可能满足所有人的需求，所以在确定总体目标时不仅要收集来自各方面的信息，包括客户、网站开发团队甚至包括浏览的用户群体，更关键的是对这些信息进行提炼，确定网站特定的任务和特定的用户群。

2）网站的用户群

一个网站成功的关键不是有好的域名，也不是做得有多漂亮，而是能够吸引客户。如果没有使用者去光顾，任何自认为再好的网站都是没有意义的。网站的成功建立在不断地了解用户、满足用户的过程中，而为了及时了解用户的需求，就必须拥有与用户有效沟通的渠道。

3）网站的功能

目前网站的功能有很多，但也不是功能越多网站就越完整。一个网站应该具备哪些功能应该与网站的总体目标一致，如一个电子商务网站，购物车是必备的功能；而一个政府网站，新闻系统却是不可少的。目前常见的网站功能有：新闻发布系统、留言板、论坛、网站计数器、Web 邮件、用户注册系统、信息下载、信息搜索、购物车、后台管理系统等。

随着 Internet 技术的发展这些功能不是一成不变的，还会有新的功能出现在网站开发中。

4）网站的响应时间

目前国内的网络传输资源极为有限，因此使用图形时一定要考虑传输时间的问题。根据经验与统计，使用者可以忍受的最长等待时间大约是 90s，如果页面无法在这段时间内传输并显示完毕，那么使用者就会毫不留情地掉头离去。因此必须依据 HTML 文件、图形文件的大小，考虑传输速率、延迟时间、网络交通状况，以及服务端与客户端的软、硬件条件，估算页面的传输与显示时间。

5）网站的易访问性

网站的易访问性是指网站能够被其访问者访问的难易程度。同类型的网站都可以通过提高网站的易访问性来改善搜索引擎对网站的索引情况，使用户更容易访问信息和获得相关的服务。

6）网站的易维护性

建站容易维护难。对于网站来说，只有不断地更新内容，才能保证网站的生命力。而许多网站使用单位的维护者不一定具备专业知识，所以网站的易维护性是网站开发人员必须掌握的因素。

三、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 1

拓展任务号	1-1	任务名称	浏览各类网站
计划用时	30 分钟	任务性质	课内
任务描述与目标			
通过对各类网站的浏览了解网页的布局			
主要操作步骤提示			
1．在 Internet 中选择不同类型的网站； 2．下载各类网站的 LOGO，分析 LOGO 的设计特点； 3．分析各类网站的页面结构，归纳总结所浏览页面的结构类型； 4．分析网站中的基本元素； 5．分析并列出不同网站的用户群； 6．分析各类网站的主要功能； 7．比较并写出不同网站的响应时间			

1.2.2 网站的要素

网站是一个综合体，一个完整的网站由多方面的元素组成。其中，网址、服务器、浏览器、网页是组成网站的主要元素。

一、实战演练

（1）打开浏览器，在浏览器的地址栏中输入网站。

（2）单击网页中的链接，观察状态栏和地址栏的变化。

二、知识点

正如前面所说的，网站开发是一项系统工程，所涉及的内容比较多。一个网站能正常运行应具备以下四个基本要素：

1. 网址

用户在浏览器中输入网址访问网站，在浏览器中看到并通过超链接进一步访问相关网页。网址是 Internet 上标示网站的地址。

1）IP 地址

IP 地址就是给每个连接在 Internet 上的主机分配的一个 32 bit 地址，通常由 4 组数字组成，中间由小圆点分隔，如 192.168.40.120。

Internet 上的每台主机都有一个唯一的 IP 地址。IP 协议就是使用这个地址在主机之间传递信息的，这是 Internet 运行的基础。IP 地址分为 4 段，每段 8 位，用十进制数字表示，每段数字范围为 0～255。IP 地址根据网络 ID 的不同分为 A、B、C、D、E 五种类型，常用的是 B 和 C 两类。不同类的地址有不同的作用。

2）域名地址

由于 IP 地址是数字型的，比较难理解和记忆，因此通常用另外一种表示方式即域名地址来表示。

域名地址的结构是一个树形结构，包括计算机名、组织机构名、网络类型名、最高层域名。因此，域名结构由若干分量组成，各个分量之间用点隔开：….三级域名.二级域名.顶级域名，如 acm.zju.edu.cn。各分量代表不同级别的域名，级别最低的域名写在最左边，级别最高的顶

级域名则写在最右边。完整的域名不能够超过 255 个字符。一个域名包含的下级域名的数目并没有明确的规定，各级域名由各自的上一级域名管理机构管理，而最高级的顶级域名则由因特网的有关机构管理。

域名地址要经过注册才能使用。最为通用的域名.com/.net 的管理机构是 ICANN，但 ICANN 并不负责域名注册，它只是管理其授权的域名注册商，如万网、新网等。每一个域名的注册都是独一无二、不可重复的。因此，在网络上，域名是一种相对有限的资源，它的价值将随着注册企业的增多而逐步为人们所重视。在网址栏里输入域名地址，由 DNS 服务器将域名地址翻译为该域名所对应的 IP 地址后，才能正常连接目标服务器。

常见的域名类型如下：

com—Commercial organizations　工、商、金融等企业

edu—Educational institutions　教育机构

gov—Governmental entities　政府部门

mil—Military　军事机构

net—Network operations and service centers　互联网络、接入网络的信息中心和运行中心

org—Other organizations　各种非盈利性的组织

cn—中国专用的顶级域名

3）URL

URL（Universal Resource Locator，统一资源定位器）用于完整地描述 Internet 上网页和其他资源地址的一种标识方法。URL 由三部分组成：协议类型、主机名、路径及文件名。通过 URL 浏览器可以访问到用户所要查询的信息资源，如 http://women.sohu.com/20100201/n269918641.shtml，其中 http 是协议类型，women.sohu.com 是主机名，/20100201/ n269918641.shtml 则是路径及文件名。

常见的协议类型如下：

✧ http：文件在 Web 服务器上；

✧ file：用于浏览本地文件，如 file://E:/ShowA.htm；

✧ ftp：在 FTP 服务器上，用于文件的上传与下载，如 ftp://home.163.com；

✧ mailto：用于发送电子邮件，如 mailto:abc@xxx.com；

✧ news：可以访问 Internet 中各种各样的新闻组，如 news:msnews.microsoft.com；

✧ Telnet：Internet 远程登陆服务的标准协议和主要方式；

URL 可以分为绝对地址和相对地址。从协议开始的 URL 称为绝对地址，如 http://women.sohu.com；从非协议开始的 URL 地址称为相对地址，如 www.baidu.com。

2. Web 服务器

Web 服务器是互联网的节点，存储、处理网络上的数据、信息，一个 Web 服务器包括两个平台：硬件和软件。

硬件平台通常就是指计算机，根据网站的信息量的大小来选择计算机类型，对于一些小型的网站，一台普通的个人电脑也可以作服务器。

软件平台包括服务器操作系统和 Web 服务器。常见服务器操作系统有 Windows NT/2000/2003/Server、UNIX、Linux；常见的 Web 服务器有 IIS、Apache、Tomcat、NetBox。

一般 Windows 系列的操作系统选择 IIS 作为 Web 服务器软件，而 UNIX 和 Linux 则常选择 Apache 或 Tomcat。

自建 Web 网站服务器不仅费用非常高，同时还需要专业的维护人员。有的企业在购买服

务器后将其托管于一些网络服务机构，而对于一些中小型网站来说，采用租用虚拟空间的方式则是比较经济的选择。

3. 浏览器

浏览器是指可以显示网页服务器或者文件系统的 HTML 文件内容，并让用户与这些文件交互的一种软件。浏览器包括微软的 Internet Explorer、Mozilla 的 Firefox、Apple 的 Safari 及国内常用的遨游等，浏览器是最经常使用的客户端程序。

4. 网页

网页是构成网站的基本元素，是能通过浏览器解释后显示出来的一类文件。网页可以笼统地分为动态网页和静态网页。静态网页文件一般以.htm 或.html 为后缀，俗称 HTML 文件；动态网页有不同后缀的网页文件，如.ASP、.ASPX、.PHP、.JSP 等。

静态网页的内容是相对固定的，如果要修改静态网页的内容，必须打开网页文件进行修改。而动态网页上的信息则会根据用户的浏览条件自动生成或更新。

静态网页直接由浏览器解释并显示；动态网页则通过网站服务器运行生成后再传送给浏览器解释并显示。如图 1-6 所示表示了动态网页和静态网页的生成过程。

静态网页的工作过程主要分为三步：

（1）浏览器（即客户端）将用户提出的访问请求发给 Web 服务器；

（2）Web 服务器响应请求，并把找到的 HTML 文件返回给浏览器；

（3）浏览器显示请求获得的页面。

动态网页的工作过程则繁杂得多，主要的工作步骤如下：

（1）浏览器将用户提出的访问请求发给 Web 服务器；

（2）Web 服务器响应请求，并向数据库发出提取数据命令；

（3）数据库得到请求后，验证请求的合法性，对数据进行处理后将处理结果返回给 Web 服务器；

（4）Web 服务器通过编译把动态页面编译成标准的 HTML 代码，传递给用户浏览器；

（5）浏览器显示请求获得的页面。

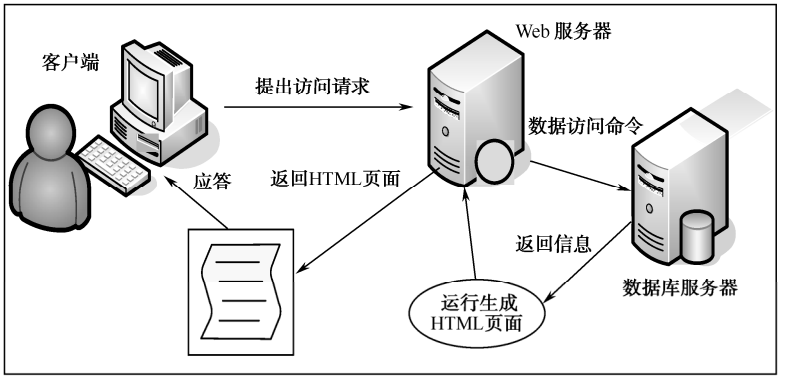

图 1-6　动态网页和静态网页的生成过程

1.3　任务 2　ASP.NET 开发环境

在当今编程领域，B/S 结构的编程方式最具代表性。.NET 框架下的 ASP.NET 和 J2EE 框架下

的 JSP 是目前最具有代表性的 B/S 结构的编程方式。相对于 JSP，ASP.NET 因其入门简单、开发方便的特点为越来越多的开发人员所认可。ASP.NET 目前最新的开发环境是 Visual Studio 2012。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）掌握 Visual Studio 2012 的安装环境、安装过程

（2）了解 Visual Studio 2012 不同版本的特点；

（3）了解 Visual Studio 2012 不同 Web 开发方式；

（4）掌握 IIS 的安装与配置；

（5）了解.NET 开发平台模型的结构；

（6）初步掌握 Visual Studio 2012 开发环境的应用；

（7）了解 ASP.NET 开发网站的基本步骤。

1.3.1 Visual Studio 2012 的安装

一、实战演练

（1）Visual Studio 2012 安装的启动界面与以往的 Visual Studio 相比，有了很大的不同。启动安装光盘或在安装光盘中找到 vs_ultimate.exe 可执行文件并双击后直接进入如图 1-7 所示的安装位置选择界面，在选择安装位置后单击“下一步”按钮，进入安装功能选择界面。根据需要选择安装内容。

图 1-7 安装位置选择界面

（2）安装完成后，会显示如图 1-8 所示的安装完成界面，这表示该软件已经成功安装在机器上。单击启动按钮会提示安装时用的激活码，输入正确的激活码就可以使用了。如果没有激活码，则可以试用一个月时间。

图 1-8　安装完成界面

（3）完成安装后，执行“开始”→“所有程序”命令，找到“Microsoft Visual Studio 2012”文件夹下的“Visual Studio 2012”并单击，就可以打开 Visual Studio 集成开发环境，也可以在桌面上创建快捷方式以方便启动。

在第一次启动 Visual Studio 2012 时，选择进入启动界面，第一次运行 Visual Studio 程序会自动配置运行环境，这时需要等待一会儿。

（4）初次启动配置后，会跳出一个初次使用的环境配置界面进行默认环境设置，环境配置界面如图 1-9 所示。根据自己的需要设置默认环境，如果使用多钟语言进行开发，则可选择【常规开发设置】，设置完毕后单击【启动 Visual Studio】启动程序。

（5）启动 Visual Studio 2012 后，开始界面如图 1-10 所示，界面的左边是“开始”和“最近”二个部分。通过“开始”下面的选项可以“新建项目”和“打开项目”。在“最近”的下方会显示最近打开过的项目。右边是一个选项卡，最初显示的是“操作方法视频”，通过选择“入门”和“最新新闻”可以学习和了解目前 Visual Studio 最新的相关技术的动态信息。

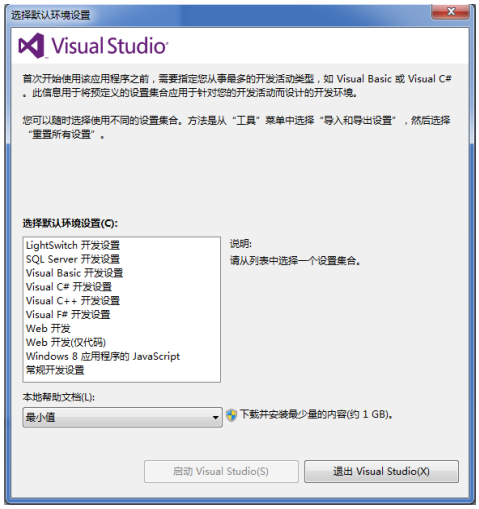

图 1-9　初次使用的环境配置界面

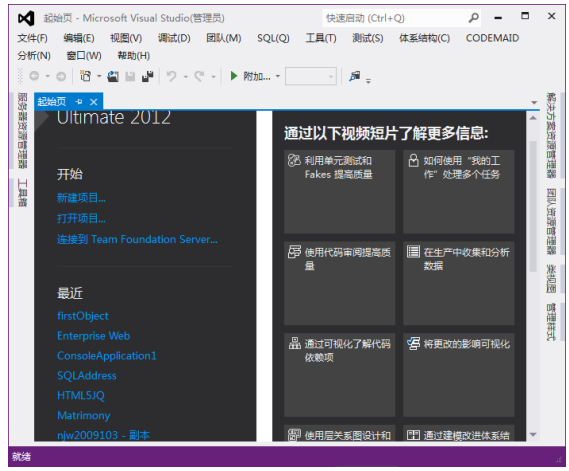

图 1-10　Visual Studio 2012 开始界面

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-1 所示。

表 1-1 演练完成情况评价表

任务号	1-1	任务名称	安装 Visual Studio 2012
任务子项	完成情况	主要问题	未完成原因
安装 Visual Studio 2012 开发环境			
上网了解 Visual Studio 2012 不同版本的性能			

三、知识点

1. Visual Studio 2012 的安装环境

Visual Studio 需要安装在 Windows 操作系统中，并且对系统的硬件性能及兼容性有一定的要求。

1）操作系统

✧ Windows 7 或者更高的版本

2）硬件要求

✧ 1.6GHz 或更快的处理器，1GB 内存，10GB 可用硬盘空间。

✧ 最小安装需要 1.22 GB 可用磁盘空间

✧ 完全安装需要 2 GB 可用磁盘空间

2. Visual Studio 2012（VS 2012）的新特性

和以前的版本相比 VS2012 最大的新特性莫过于对 Windows 8 Metro 开发的支持。

VS2012 集成了 ASP.NET MVC 4，全面支持移动和 HTML5。

VS2012 支持.NET 4.5，和.NET 4.0 相比，.NET 4.5 更多的是完善和改进，.NET 4.5 也是 Windows RT 被提出来的首个框架库。

VS2012+TFS2012 实现了更好的生命周期管理，VS2012 不仅是开发工具，也是团队的管理信息系统。

3. Visual Studio 2012 的不同版本

Visual Studio 2012 开发团队每次都会发布多个正式版本，根据不同的团队需求和规模及其成员的不同角色量身定制的。通过访问 MSDN 网站可以选择自己所需要的版进行安装，每个版本都有一个月的试用期。Visual Studio 2012 正式发布的版本一共有三个，每版本的安装文件各不相同。

1）旗舰版（Ultimate）

Visual Studio Ultimate 2012 是全面的 ALM 与敏捷开发解决方案，可供开发高度可扩展的软件应用程序并经营相关服务的中大型企业企业使用。安装程序 vs_ultimate.exe。

2）高级专业版（Premium）

包括敏捷项目规划和管理、利益干系人和用户参与、开发人员工作效率以及质量实现和测试功能，可以提供集成的 ALM 与敏捷开发解决方案。向业务分析员、项目经理和测试人员以及开发人员和用户提供了可实现无缝流程整合和协作的工具。安装程序 vs_premium.exe。

3）专业版（Professional）

专业开发人员需要专业工具，为开发人员带来了统一的开发体验，使他们可以创建跨Web、云和设备的多层应用程序。安装程序 vs_professional.exe。

4. Visual Studio2012 的获取

从Microsoft站点 www.microsoft.com/express/上可以下载Visual Studio Express 2012 for Web的免费版本。也可以从 Microsoft 的站点 http://www.microsoft.com/zh-cn/download 上下载有免费试用期的版本。如果刻录到 DVD 或使用虚拟光驱，则可以选择下载一个 ISO 映像程序，或者选择下载 Web 安装程序。

1.3.2 Visual Studio 2012 Web 开发环境

一、实战演练

（1）选择合适的文件位置，创建一个文件夹“firstObject”。

（2）打开“开始”→“所有程序”→“Microsoft Visual Studio 2012”→“Visual Studio 2012”并单击，就可以打开 Visual Studio 集成开发环境，也可以在桌面上创建快捷方式以方便启动。

（3）在开发环境的菜单栏中选择“文件”→“新建”，“新建”菜单栏下有 2 种项目创建方式：网站（web site）、项目（web application）。选择“网站”，打开如图 1-11 所示的“新建网站”对话框。“新建网站”对话框的最上面选择“.NET Framework 4.5”的支持环境，选择“Visual C#”/“ASP.NET 空网站”模板，在下面的“Web 位置”选择“文件系统”，单击“浏览”按钮，在弹出的“选择位置”对话框中选中第（1）步中创建的文件夹，出现如图 1-11 所示的网站路径。单击“确定”按钮。

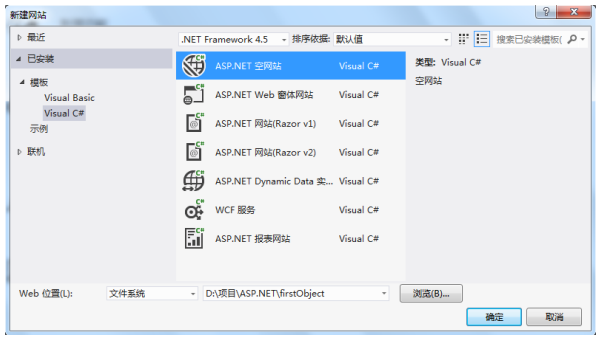

图 1-11 新建网站对话框

（4）新建网站后的 Visual Studio 2012 开发环境如图 1-12 所示。在开发环境的右边是“解决方案资源管理器”，在管理器中列出了新建项目名“firstObject”，以及自动创建的文件“Web.config”。

（5）在“解决方案资源管理器”中右击“firstObject”项目名，在弹出的快捷菜单中选择“添加”→“添加新项”，或是在下拉菜单中选择“文件”→“新建”→“文件”，弹出如图 1-13 所示的“添加新项”对话框。在对话框中选择“Visual C#” →“Web 窗体”，名称就使用“Default.aspx”，选择“将代码放在单独的文件中”，单击“添加”按钮。

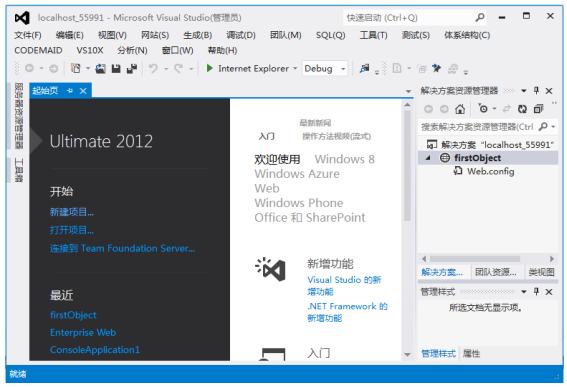

图 1-12　Visual Studio 2012 开发环境

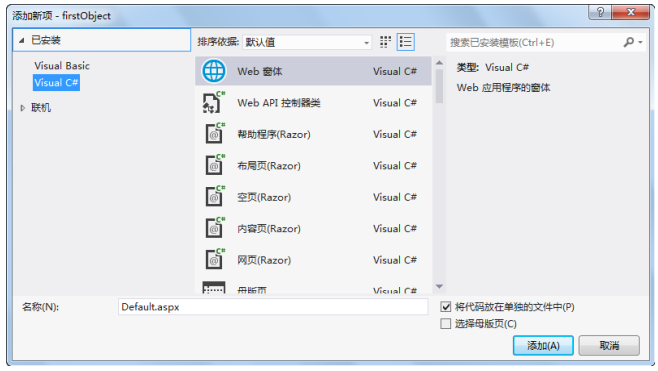

图 1-13　“添加新项”对话框

（6）在开发环境中的左边是“工具箱”，中间是“工作窗口”。“工作窗口”是一个可见即可得的设计工具，可以通过下方的“设计”/“源”/“拆分”选项选择不同方式的工作窗口。在“工作窗口”代码中添加“<title>我的第一个网站</title>”和“Hello World!”，添加文件“Web窗体”→“Default.aspx”开发环境如图 1-14 所示。

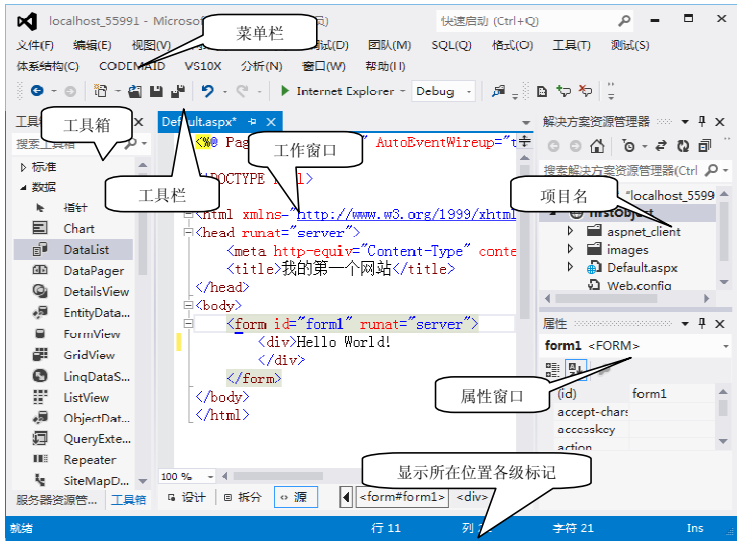

图 1-14　添加新项后的开发环境

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-2 所示。

表 1-2　演练完成情况评价表

任务号	1-2	任务名称	在项目中添加 Web 页面
任务子项	完成情况	主要问题	未完成原因
在“firstObject”添加一个新的 Web 页面			
熟悉 Visual Studio 2012 开发环境			

三、知识点

Visual Studio 集成了 ASP.NET Web 应用程序、桌面应用程序、移动应用程序和 XML Web Services 等不同类型的应用程序的开发。可以用 Visual C#、Visual Basic、Visual C++、Visual F# 和 JavaScript 编写代码，可以更轻松地创建混合语言解决方案。

1. Visual Studio 2012 的 Web IDE 设置

打开 Visual Studio 后，可以看到整个 IDE 包括许多工具窗口、菜单和工具栏，以及主窗口空间。工具窗口停靠在应用程序窗口的左侧和右侧，其顶部有“快速启动”、菜单栏和标准工具栏。应用程序窗口的中心是“起始页”，在解决方案或项目加载后，编辑器和设计器会出现在这里，开发应用程序时，大部分时间将在这里的中央区域工作。开发者可以根据开发阶段的工作性质打开、关闭这些窗口，使开发环境符合开发者的最佳工作状态。Visual Studio 允许开发者根据自己的偏好定制 IDE。有时可能需要把窗口重新安排到容易找到的位置，有时可能希望打开其他经常用到的窗口。Visual Studio 是可以完全自定义的，而且 IDE 的任何细节都可以进行调整。

1）IDE 窗口布局

通过“视图”菜单可以打开开发者所需要使用的工具、对话框和设计器。打开的工具窗口有“自动隐藏”、“停靠”、“浮动”三种状态。可以通过窗口上方的自动隐藏按钮、窗口位置选择按钮和关闭按钮来改变窗口状态，窗口位置按钮是一个下拉菜单。

拖动对话框窗口上方的标题栏，可以把窗口由“停靠”变为“浮动”，也可以通过“菱形向导”的工具把“浮动”窗口拖放到指定位置。通过向导，可以清楚地看到窗口将如何停靠。只需拖动窗口的标题栏，将其拖动到菱形向导中的某个预览图即可。处理“拖动”状态的开发环境如图 1-15 所示。

2）IDE 的环境设置

IDE 的环境设置可以通过使用“选项”对话框对 Visual Studio 进行其他自定义操作，打开“工具”→“选项”可以打开如图 1-16 所示的“选项”对话框。在“选项”对话框可以更改编辑器中文本的字体和字号，或者更改 IDE 的主题颜色等。开发者可以根据自己的需要对 IDE 进行自定义设置。

2. ASP.NET 概述

.NET 框架是微软提供的一种开发模型，是一个多语言的组件开发和执行环境，它提供了一个跨语言的统一编程环境。在生成中间语言之前，各种语言利用各自编程语法进行应用开发，

在生成中间语言之后，各个语言可能对中间语言进行相互调用，实现框架内应用的重复利用。C#语言是随着.NET 平台一起发布的一个新的面向对象的语言，是现今微软平台上最重要的语言。.NET 框架由不同的组件组成，这些组件有助于创建和运行基于.NET 的应用程序。.NET 框架如图 1-17 所示。

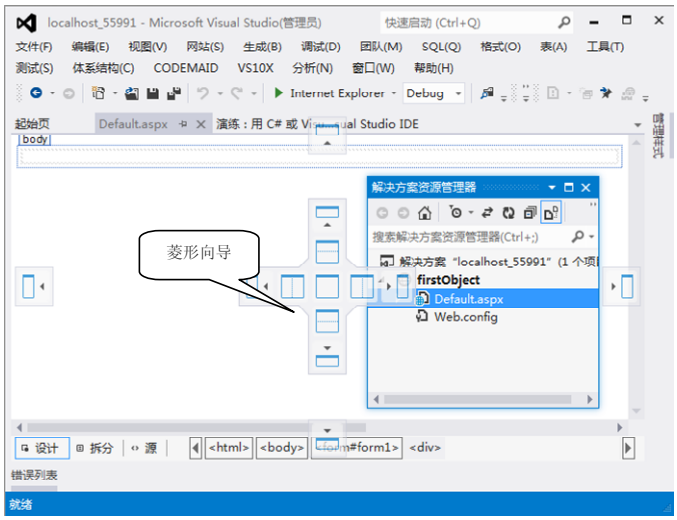

图 1-15　处理“拖动”状态的开发环境

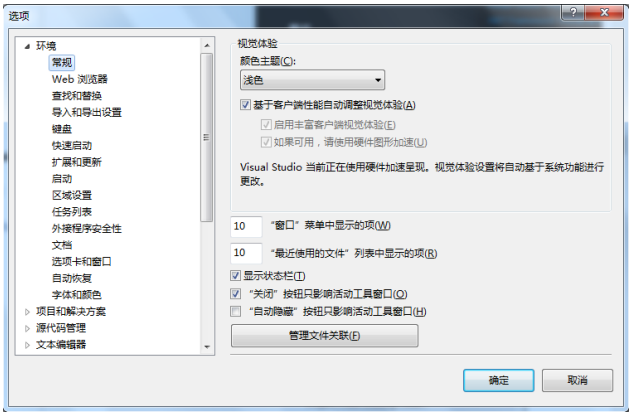

图 1-16　“选项”对话框

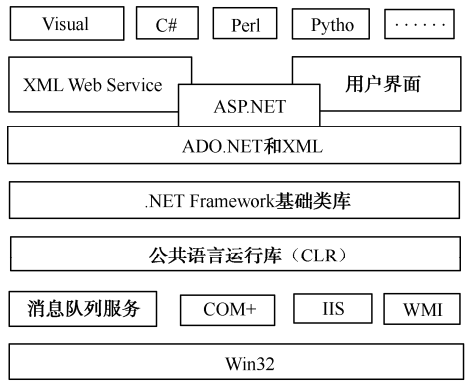

图 1-17　.NET 框架

2002 年初首次发布.NET Framework 1.0 以来，ASP.NET 就是.NET Framework 重要的一部分，可以用来构建富 Web 应用程序。与传统 ASP 相比，ASP.NET 1.0 及相关的 Visual Studio .NET 的引入使得页面显示与代码清楚地分开、开发模型更接近于桌面应用程序的编程方式等优势，同时有了一个功能丰富的开发工具——Visual Studio .NET，开发人员可以通过它可视化地创建和编写 Web 应用程序代码。在 2003 年发布了.NET Framework 的另一个版本(称为.NET 1.1)和开发 IDE(称为 Visual Studio .NET 2003)。

2005 年 11 月，Microsoft 发布了 Visual Studio 2005 和 ASP.NET 2.0。让全球许多开发人员感到惊喜的是，Microsoft 又大大改进和扩展了产品，增加了许多功能和工具来帮助降低 ASP.NET 1.0 所带来的复杂性。新的向导和智能控件减少了构建应用程序所需的代码，降低了新开发人员的学习难度，并且提高了开发效率。

2007 年 11 月，Microsoft 发布了 Visual Studio 2008 和 ASP.NET 3.5 框架。

在 2010 年 3 月发布了 Visual Studio 2010 和 ASP.NET 4，这两个版本都增加了许多新功能，包括 LINQ、AJAX 框架整合、ADO.NET Entity Framework 和 jQuery。

2012 年 9 月正式发布了 Visual Studio 2012 和 ASP.NET 4.5，它是在已成功发行的 Visual Studio 2010 和 ASP.NET 4 基础之上构建的，它提供了新的模板、设计工具以及测试和调试工具——在尽可能短的时间内构建具有强大吸引力的应用程序所需要的一切； Blend for Visual Studio 还提供了一款可视化工具集，可以充分利用 Windows 8 全新而美观的界面；利用 Team Foundation Service，无需基础架构，就可以进行应用生命周期管理。

3. Web 项目类型

在 Visual Studio 2012 中，可以为创建 ASP.NET Web Forms 网站选择两种项目类型：Web Application 项目和 Web Site 项目。

1）Web Site 项目类型

Web Site 项目是在 Visual Studio 中创建的 Web 站点方式。创建新的 Web Site 项目的方法是：选择“文件”→“新网站”命令，或者选择“文件”→“新建”→“网站”命令。

Web Site 项目站点只是一个文件夹，以及其中的一组文件和子文件夹。在该 Web 站点中没有跟踪所有单个文件的项目文件（扩展名为.csproj）。只需将 Visual Studio 指向一个文件夹，它就会一直把这个文件夹作为 Web 站点打开。这样就可以非常容易地创建站点的副本、移动它们或者与别人共享，因为它不依赖于本地系统中的文件。

2）Web Application 项目类型

Web Application 项目类型可以让主团队开发人员，以及那些需要对站点内容和编译及部署过程有更多控制的开发人员更容易使用 Visual Studio 构建 Web 站点。 整个 Web 站点作为一个项目进行管理，用单个项目文件跟踪 Web 站点的所有内容。

在 Visual Studio 中，可以通过“文件”→“新建”→“项目”对话框来创建一个新的 Web Application Project。在该对话框中，单击首选编程语言(可以是 Visual Basic 或 Visual C#)，然后单击 Web 类别，其中有若干个 ASP.NET Web 应用程序模板。其中一个可用的项目模板是 ASP.NET MVC 4 Web Application，它是根据模型→视图→控制器(Model View Controller，MVC)模式来创建应用程序的，该模式是 Web 应用程序开发的另一个流行方式。

3）如何选择项目类型

由于有两个选项可供选择，那么应选择哪个项目类型呢？一般来说，Web Site Project 更容易使用，因为它只是一个文件夹，所以更容易把文件复制到另一个位置，例如另一个开发工作站或生产服务器。另外，对代码文件的修改会由 Web 服务器提取，并自动应用，无需正规的部署过程。

另一方面，如果一组开发人员在同一个站点上工作，Web Application Project 就比较好，因为它具备更正式的开发和部署过程，更好地支持源控制版本系统，例如 Microsoft 的 Team Foundation Server。

Web Application Project 模板与使用 Web Site Project 模板有很多共同之处，本教材中使用的是 Web Site 项目模板，这是因为该模板对于 ASP.NET 初学者来说很容易使用。

4. Web Site 项目模板类型

在如图 1-11 所示 Visual Studio 的新建网站对话框中包含不同的 Web 站点模板，各个模板

的用途各不相同。在对话框的左侧部分，可以从 Visual Basic 和 Visual C#中选择一种作为站点的编程语言。中间的部分显示了默认安装的 ASP.NET Web 站点模板。本教材涉及的项目都采用 ASP.NET 空网站模板，其他的模板只作简单的介绍。

1. ASP.NET Web 窗体网站

这个模板允许配置一个基本的 ASP.NET Web 站点。它包含许多文件和文件夹用于开始站点的开发。

2. ASP.NET 网站(Razor v1 或 Razor v2)

通过 Microsoft 的 Web Pages 框架，使用这些模板可以创建站点。

3. ASP.NET 空网站

ASP.NET 空网站模板只包含一个配置文件(Web.config)。如果包含用于创建新的 Web 站点或希望从头开始创建站点，则 ASP.NET 空网站模板会很有用处。

4. ASP.NET Dynamic Data 实体网站

这个模板用于创建灵活且强大的 Web 站点来管理数据库中的数据，而不需要手动输入许多代码。

5. WCF 服务

该模板可用于创建包含一个或多个 WCF(Windows Communication Foundation) Service 的 Web 站点。WCF Service 模板与 Web Service 模板有点相似，它用来创建可通过网络调用的方法。然而，WCF Services 比这个简单 Web 服务要复杂得多，而且提供了更大的灵活性。

没有必要过多考虑如何正确选择的 Web 站点模板。在 Visual Studio 中， ASP.NET Web 站点实质上只是对文件夹的一个引用，因此很容易将一个模板的类型添加到另一个模板上。例如，向标准的 ASP.NET Web 窗体网站或 ASP.NET 空网站中添加 Web 服务文件，是完全可以的。

6. Visual Web Developer 网站类型

Visual Web Developer 是 Visual Studio 开发平台上的一个重要组成部分，是一种用于在多种配置中创建和使用 ASP.NET 网站的一组实用工具，利用它可以创建多种类型的项目。可以创建配置的 ASP.NET 网站有：文件系统站点、本地 IIS 站点、文件传输协议部署的站点和远程站点。一般来说，开发时基本采用文件系统站点的方式进行。

1）文件系统站点

在文件系统网站中，可以在任何所需的文件夹中创建和编辑文件，其位置可以在本地计算机上或是在通过网络共享访问的另一台计算机上的文件夹中，无需在计算机上运行 IIS。可以使用 ASP.NET Development Server 来测试网页，但不能向其他计算机提供网页。因此，它只适用于在本地测试网页。

2）FTP 部署的站点

通过 Visual Studio，可以打开和编辑 FTP 服务器上可用的网站。网站位于宿主站点上，这是一种典型方案。可以从 Visual Studio 内部连接到对其具有读/写权限的任何 FTP 服务器，然后，可以在该服务器上创建和编辑网页。如果 FTP 服务器配置有 ASP.NET 和一个指向 FTP 目录的 IIS 虚拟根目录，则还可以从该服务器运行网站对其进行测试。

3）本地 IIS 站点

本地 IIS 站点方式可以逼真地模拟网站在成品服务器中如何运行，相对于使用 ASP.NET Development Server 运行的文件系统网站，这更具有优势，因为路径将按照其在成品服务器上

的方式解析。使用这种方式建立网站时必须在本地计算机上安装 IIS。

7. Web 网站基本文件

ASP.NET 4.5 Web Forms 站点至少由一个 Web 窗体(扩展名为.aspx 的文件)组成，但是它常常是由更多文件组成的。VS 中有许多不同的文件类型可用，各个类型提供了不同的功能。下面简单介绍 VS 中用到的最重要的文件类型，以及添加这些文件的方法。

1）Web 文件

Web 文件是 Web 应用程序中特有的文件，可以由浏览器直接请求，也可以用来构建在浏览器中请求的 Web 页面的一部分。下面列出了在 ASP.NET Web Forms 网站中常用的各种 Web 文件和它们的扩展名，并说明了各种文件的用法。

Web Form 文件扩展名为.aspx，这类文件是所有 ASP.NET Web 站点最重要文件，是用户在浏览器中浏览的页面。

✧ 母版页文件。扩展名为.master　允许定义 Web 站点的全局结构和外观；

✧ 用户控件文件。扩展名为.ascx　包含可在站点的多个页面中重用的页面片段；

✧ HTML 页面文件。扩展名为.htm/ .html　可用来显示 Web 站点中的静态 HTML；

✧ 样式文件 扩展名为.css。包含允许定制 Web 站点的样式和格式的 CSS 代码；

✧ Web Configuration 文件。扩展名为.config　包含用在整个站点中的全局配置信息；

✧ 脚本文件　扩展名为.js。包含可以在客户端浏览器中执行的 JavaScript。

2）功能代码文件

代码文件分为三种，主要负责 Web 项目中逻辑功能设计。代码文件包括：

✧ WCF 服务文件。扩展名.svc，可以被其他系统调用，包括浏览器，可以包含能在服务器上执行的代码。

✧ 类文件。扩展名.cs/vb，包含构建 Web 站点的代码。 注意隐藏代码文件也有相同的扩展名， 因为它们实质上是类文件。 C#使用带有.cs 扩展名的文件，而 Visual Basic 使用的是带有.vb 扩展名的文件。

✧ Global 全局应用程序类文件。扩展名.asax，可以包含为了响应站点中触发的代码，例如应用程序的开头或者当在站点中某处发生错误时。

✧ 一般处理程序。扩展名.ashx， 一个 httpHandler 接受并处理一个 HTTP 请求的文件。

3）数据文件

数据文件用来存储可以用在站点和其他应用程序中的数据。这组文件由 XML 文件、数据库文件以及与使用数据相关的文件组成。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 2

拓展任务号	1-2	任务名称	安装 Visual Studio 2012
计划用时	60 分钟	任务性质	课内
任务描述与目标			
安装 Visual Studio 2012 并了解其版本情况			
主要操作步骤提示			

续表

拓展任务号	1-2	任务名称	安装 Visual Studio 2012
计划用时	60 分钟	任务性质	课内
1．完成安装 Visual Studio 2012； 2．完成安装 MSDN； 3．打开 Visual Studio 2012 并新建一个网站，写出开发环境的主要窗口，练习各主要窗口的打开、关闭和选择停靠方式； 4．打开 MSDN，在展开的“开发工具与语言”/“Visual Studio 文档”结点，找到“Visual Web Developer”结点，了解 MSDN 关于 Web 开发的帮助文档			

1.3.3 IIS 的安装与配置

IIS 是 Windows 操作系统服务器软件，Windows 7 旗舰版一般都支持 IIS。

一、实战演练

IIS 是 Internet Information Service 的简称。执行“控制面板”→“管理工具”命令，查看是否有“Internet 信息服务（IIS）”的图标，如果没有，就先安装 IIS，安装步骤如下：

（1）选择“控制面板”→“程序”，打开程序控制面板，如图 1-18 所示。

（2）在程序控制面板中选择“打开或关闭 Windows 功能”。出现如图 1-19 所示的“打开或关闭 Windows 功能”功能面版，在这个面版中对 IIS 进行添加。注意，如果你的 Web 程序是用.NET 开发的，请务必选中“ASP.NET”复选框。

图 1-18 程序控制面板

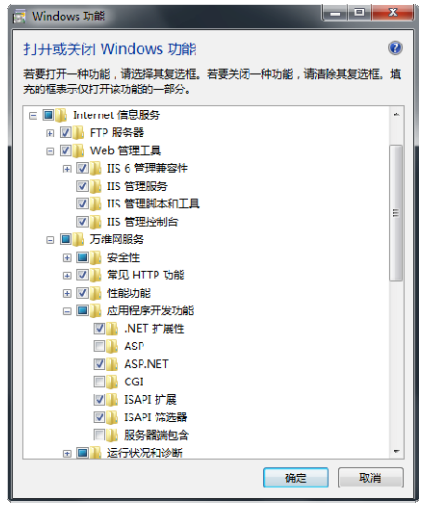

图 1-19 “打开或关闭 Windows 功能”对话框

（3）安装完成后，选择“控制面版后”→ “系统与安全”→“管理工具”，在管理工具面版中选择。打开如图 1-20 所示的 IIS 控制面版。

（4）打开 IIS 面版后可以看到 IIS 服务器有“应用程序池”和“网站”，为了管理网站方便，可以先添加 2 个“应用程序池”，一个用于管理.NET4.0，另一个用于管理.NET 2.0。右击“应用程序池”，打开如图 1-21 所示的“添加应用程序池”对话框。根据需要选择相应的.NET

Framework 版本。

图 1-20　IIS 信息服务管理器

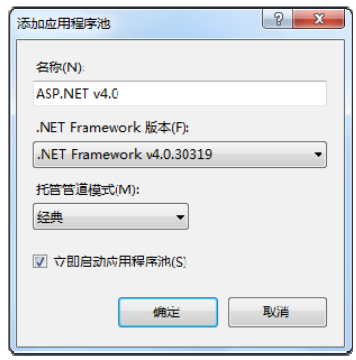

图 1-21　“添加应用程序池”对话框

（5）右击网站“添加网站”，打开如图 1-22 所示的“添加网站”对话框。在“应用程序池”选择上一步设置的应用程序池。如果不选择 IIS 会自动设计与网站名称相同的应用程序池，然后再到应用程序中去更改应用程序池的.NET FrameWork 对应的版本。

（6）打开“$\Windows\System32\drivers\etc”文件夹下“hosts”文件，用记事本打开，在记事本中输入“127.0.0.1　www.firstObject.com”。设置后的“hosts”文件如图 1-23 所示。

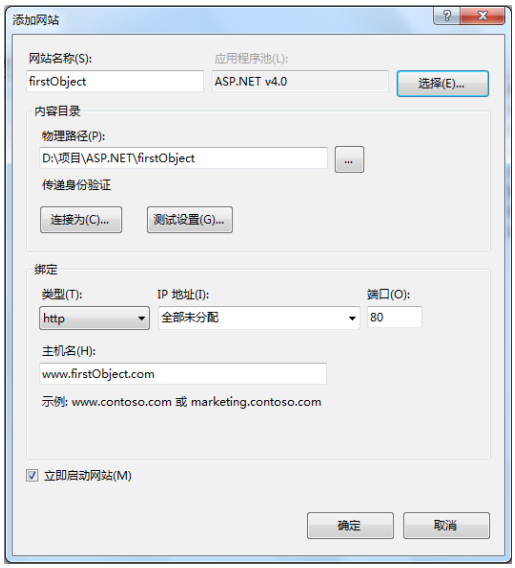

图 1-22　“添加网站”对话框

图 1-23　“hosts”文件配置

（7）在浏览器中输入 http://www.firstobject.com，在浏览器中查看浏览效果。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-3 所示。

表 1-3 演练完成情况评价表

任务号	1-3	任务名称	IIS 的安装与配置
任务子项	完成情况	主要问题	未完成原因
安装 IIS 信息服务组件			
配置一个新网站			

三、知识点

如果是在刚刚安装好的 IIS 上新建一个网站，不是所有的配置都能一次就能成功的。一般来说有三个原因。

1. 先安装 Visual Studio 再安装 IIS

使用本地 IIS 站点方式建立网站，最好先安装 IIS 后再安装 Visual Studio，如果先安装了 Visual Studio 再安装 IIS，一般都会出错，不能正常使用。在安装 Visual Studio .NET 之后，会创建 Internet 信息服务（IIS）映射，以便为 ASP.NET 的新文件扩展名及设置建立关联。如果在运行 SDK 或 Visual Studio 安装程序时没有安装 IIS，或是在运行 SDK 或 Visual Studio 安装程序之后卸载并重新安装了 IIS，那么就会使 IIS 失去与 ASP.NET 的映射，试图查看 ASP.NET 页时会遇到意外现象。因此，要为 ASP.NET 修复 IIS 映射，修复步骤如下：

（1）执行“开始”→“所有程序”→“Microsoft Visual Studio 2012”→“Visual Studio Tools”→“Visual Studio 2012 开发人员命令提示”命令，打开命令提示界面。

（2）在命令提示符下输入命令“Aspnet_regiis-i”然后按【Enter】键。运行情况如图 1-24 所示。

2. 项目文件夹的权限不够

一般来说 Windows 7 对文件及文件夹的权限管理是比较严格的，如果系统赋予操作权限不够，网站也不能够正常运行。打开网站所在的物理文件夹，右击选择“属性”，打开“文件夹”属性对话框，选择“安全”选项，单击“编辑”按钮打开“权限”设置对话框。具体设置如图 1-25 所示。

图 1-24 “Aspnet_regiis-i”命令执行界面

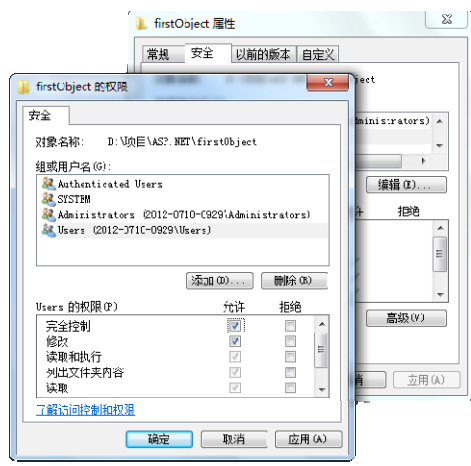

图 1-25 文件夹权限设置对话框

3. 应用程序池没有启动“ISAPI 和 CGI 限制”项

利用 ISAPI 和 CGI 限制，可以控制是否提供动态内容。当出现的错误：“HTTP 错误 404.2-Not Found”，是由于 Web 服务器上的“ISAPI 和 CGI 限制”列表设置，无法提供您请求的页面。此时需要对 IIS 的“ISAPI 和 CGI 限制”进行设置。

打开 IIS 管理器，选择 IIS 根节点，导航到“IIS”列表，选择“ISAPI 和 CGI 限制”项，如图 1-26 所示。双击“ISAPI 和 CGI 限制” 项，右击禁止的 DotNet 版本，把该项设置为允许即可，设置如图 1-27 所示。

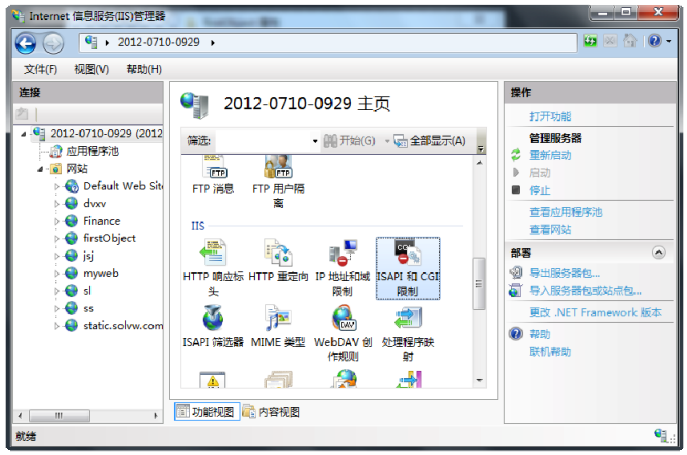

图 1-26　IIS 管理器

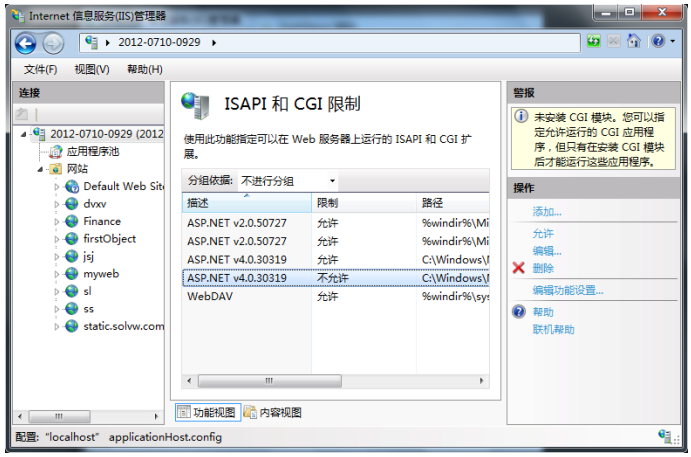

图 1-27　ISAPI 和 CGI 限制设置

1.4　任务 3　ASP.NET Web 页面

任务描述

通过一个开发环境中所介绍的简单网站完成了前面的学习任务，对 ASP.NET 的网站技术开发有了一定的认识。本任务是创建一个简单的 ASP.NET Web 应用程序，用户在网页的文本

框中输入自己的姓名后，网页上会显示出该用户的问候语。

解决方案

为完成本项目，需要完成以下任务：

（1）建立 Web 可视页面；

（2）创建事件处理程序；

（3）网站的调试与发布。

1.4.1 Web 可视页面

一、实战演练

（1）打开 Visual Studio 集成开发环境，选择“文件”→“网站”，在打开的网站对话框中选择“文件系统”，找到“firstObject”文件夹打开。在“解决方案管理器中”打开“Default.aspx”文件。

（2）选择“设计”模式，为页面添加背景图片。

为了美化网站，首先用绘图软件设计网站的背景图片。为了网站的结构合理，在“解决方案资源管理器”窗口，右击项目名“firstObject”，在弹出的快捷菜单中选择“新建文件夹”选项，并把新建的文件夹改名为“images”，同时把所做的背景图片存放到这个文件夹中，如图 1-28 所示，设置背景图片中可以看到解决资源管理器的变化。

（3）在“设计”窗口中，选择“ ”标记，在开发环境右下角的“属性”面板中选择“Style”属性，单击[...]按钮，弹出“修改样式”对话框，如图 1-28 所示，在该对话框中对“字体”、“背景”和“定位”三个选项卡进行设置。

在“源”视图中可以看到样式修改后的代码，代码如下：

```
    <div style="width: 758px; height: 188px; background-image: url('images/
xml.jpg');font-family: 华文新魏; font-size: 18px;">
    </div>
```

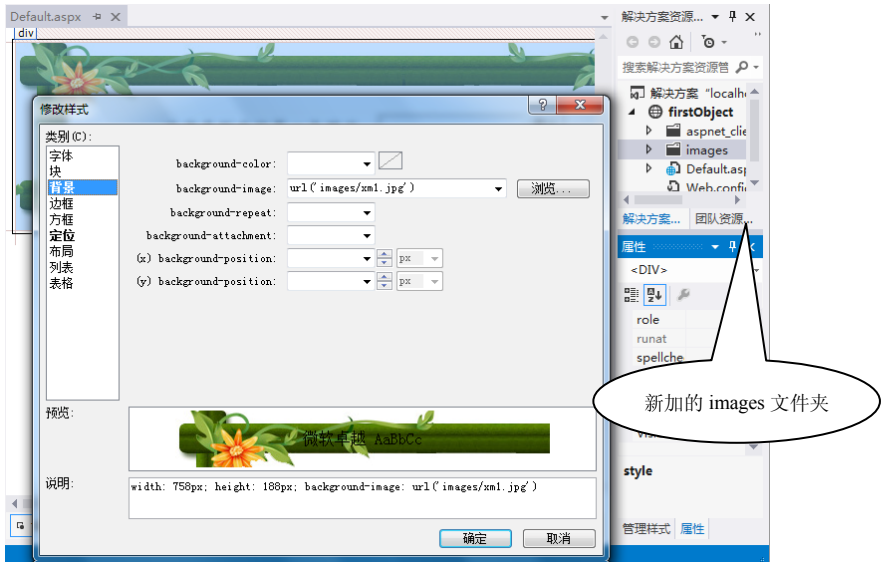

图 1-28 设置背景图片及“修改样式”对话框

（4）在位于开发环境左边的工具箱中选择并展开“HTML”选项卡，选择“div”选项并按住不放，拖曳到设计窗口中的第一个“div”中。在“属性”面板中，打开“修改样式”面板，修改布局，这时可以对当前的“div”采用拖曳方式进行定位和大小修改。切换到“源”视图，在代码中添加 id 和 runat 两个属性。修改后的代码如下：

```
    <div id=" well1" runat="server" style=" width: 460px; height: 38px;margin:70px
0px 0px 147px; ">
    </div>
```

（5）在“well1”中输入文字“欢迎来到我的第一个网站”。

（6）在工具箱中选择“标准”选项卡，分别把“TextBox”和“Button”这两个服务器控件拖入“well1”中。

（7）在“属性”面板中把拖入“Button”的“Text”属性值设置为“确定”。为了保证网页文字的整体性，执行“格式”→“新建样式”命令，打开如图 1-29 所示的“新建样式”对话框，打开“选择器”下拉框，选择“input”，在“类别”中选择“字体”，具体设置如图 1-29 所示。

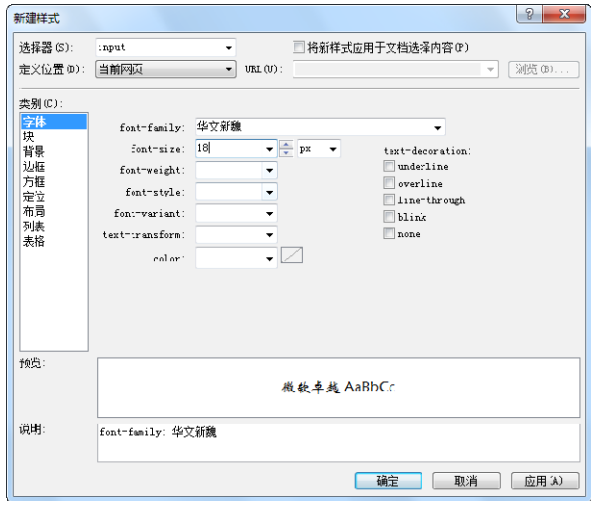

图 1-29　“新建样式”对话框

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-4 所示。

表 1-4　演练完成情况评价表

任务号	1-4	任务名称	简单页面设计
任务子项	完成情况	主要问题	未完成原因
添加页面元素			
样式设计			

三、知识点

1. Visual Studio 2012 中的 CSS

Visual Studio 2012 提供了非常方便的级联样式表（CSS）设置功能，使用它可以轻松实现

对网页的美化和排版并为设置这些功能提供了一组工具，可用来创建、应用、管理样式和级联样式表。这些工具包括：

✧ “应用样式”面板用于创建、修改和应用样式，还可以链接到或移除外部 CSS。此面板可标识样式类型，并显示该样式是否用于当前网页，以及是否由当前选定内容使用；

✧ “管理样式”面板中的很多功能与“应用样式”面板中的相同。但可以使用“管理样式”面板将样式从内部样式表（页面中的 style 元素）移到外部样式表，或从外部样式表移到内部样式表，还可以使用此面板在样式表内移动样式；

✧ “CSS 属性”面板显示网页中当前选定内容使用的样式。该面板还显示样式的优先级。此外，该面板提供所有 CSS 属性的完整列表，这样就可以向现有样式添加属性、修改已设置的属性及创建新的内联样式；

✧ “直接样式应用”工具栏，可通过它应用或移除基于类或 ID 的样式，并创建和应用新的样式。它还对 Visual Studio 生成的样式提供更大程度的控制；

✧ 标记选择器，用于在处理网页时选择 HTML 标记。它位于编辑窗口的底部。当把鼠标指针置于页面中的任何位置时，快速标记选择器栏将显示标明该区域的用基础 HTML 标记的那些标记。还可以通过按【Esc】键上移 HTML 的层次结构，以选择嵌套在其他标记内的标记。

在 Visual Studio 中，使用设计器创建和编辑网页时，可以编写不同的样式规则，如内联样式规则、包含在网页中的样式规则或包含在外部样式表中的样式规则，可以使用可视辅助查看应用于页面元素的填充和边距，还可以使用定位工具将元素定位。样式面板可以通过下拉菜单“视图”打开，如图 1-30 所示，在面板中可以通过面板下方的选项卡选择相应不同的功能面板。

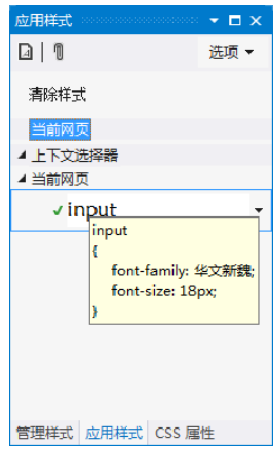

图 1-30 “样式”面板

2. Web 窗体的组成元素

ASP.NET 网页的扩展名为.aspx，为 Web 应用程序提供用户界面，ASP.NET 网页在任何浏览器或客户端设备中向用户提供信息，并使用服务器端代码来实现应用程序逻辑。

ASP.NET 网页由两部分组成：

（1）可视元素，包括标记、服务器控件和静态文本；

（2）页的编程逻辑，包括事件处理程序和其他代码。

ASP.NET 提供了两个用于管理可视元素和代码的模型，即单文件页模型和代码隐藏页模型。这两个模型功能相同，两种模型中可以使用相同的控件和代码。

在本项目的“解决方案资源管理器”中，单击文件 default.aspx 前面的“+”按钮，可以看到还有一个文件 default.aspx.cs，这是 default.aspx 的代码隐藏页，在 default.aspx 中是可视元素，页的编程逻辑则放在了 default.aspx.cs 这个文件中。

如代码清单 1-1 所示演示了新建 regist.aspx 页面，它包含组成 ASP.NET 网页的基本元素，页面包含 HTML 页中可能含有的静态文本，以及特定于 ASP.NET 的元素。

代码清单 1-1 ASP.NET 网页的源代码

```
    <%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="_Default" %>
    <!DOCTYPE html>
    <html xmlns="http://www.w3.org/1999/xhtml">
      <head runat="server">
```

```
    <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
     <title>我的第一个网站</title>
     <%--样式--%>
     <style type="text/css">
         input
         {
             font-family: 华文新魏;
             font-size: 18px;
         }
     </style>
 </head>
 <body>
     <form id="form1" runat="server">
       <div style="width: 758px; height: 188px; background-image: url('images/
xm1.jpg');
             font-family: 华文新魏; font-size: 18px;">
       <div id="wel1" runat="server" style="width: 460px; height: 38px; margin:
70px 0px 0px 147px;">
         欢迎来到我的第一个网站，
          <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
          <asp:Button ID="Button1" runat="server" Text="确认" OnClick="
Button1_Click" />
       </div>
        </div>
     </form>
 </body>
 </html>
```

在浏览器中只能看到.aspx 文件中的可视元素。ASP.NET 页所包含的元素类型如表 1-5 所示。

表 1-5　ASP.NET 页所包含的元素类型

元　素	描　述
指令	指定页属性和配置信息，最常用的指令如例中@ Page 指令
静态的 HTML 标记	标准的 HTML 元素，在 ASP.NET 中作为静态控件处理，可以在客户端浏览器中显示
HTML 注释	<!-- -->，允许在页面中添加解释性文字，不会在浏览器中显示
服务器端代码	处理页面时在服务器上运行的代码，位于< script >代码声明块或<% %>呈现块中，ASP.NET 支持多种语言
事件处理器	单文件模式中，用于处理页面或服务器控件的事件
<script>代码声明块	单文件页模型，编程代码位于<script>块中
<% %>呈现块	嵌入式代码块是在呈现页面的过程中执行的服务器代码，仅在呈现页的过程中执行
客户端<script>块	用于存放客户端上执行的脚本代码
服务器端注释	<%-- --%>允许在页面中添加解释性文字，与 HTML 注释不同，这种文本不发送到客户端
ASP.NET 服务器控件	允许用户与页面交互的控件，包括按钮、文本框、列表等。这些 Web 服务器控件与 HTML 按钮和 input 元素类似，但这些控件在服务器上处理时，允许使用服务器代码对其属性进行设置。这些控件还可以引发在服务器代码中进行处理的事件
用户控件	用于 ASP.NET 网页的相同技术创建可重复使用的自定义控件，是在扩展名为.asxc 的文件中声名定义的，为 UI 及与 UI 相关的代码重用提供了一个简单机制
自定义服务器控件	是 ASP.NET 的另一种重用机制，在类文件中定义并在使用前预编译成托管程序集

3. 指令

ASP.NET 页通常会包含一些指令，指令的基本格式如下：

```
    <%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs"
Inherits="Default" %>
```

其中最常用的为 @Page 指令，@Page 指令只能在 .aspx 文件中使用。指令后可以指定多个属性及属性值。常用的@ Page 属性如下：

✧ AutoEventWireup 属性：指示页的事件是否自动绑定。如果启用了事件自动绑定，则为 true，默认值为 true；

✧ Buffer 属性：确定是否启用了 HTTP 响应缓冲。如果启用了页缓冲，则为 true，默认值为 true；

✧ CodeFile 属性：指定指向页引用的代码隐藏文件的路径。此属性与 Inherits 属性一起使用可以将代码隐藏源文件与网页相关联，此属性仅对编译的页有效；

✧ ContentType 属性：将响应的 HTTP 内容类型定义为标准的 MIME 类型。支持任何有效的 HTTP 内容类型字符串；

✧ Culture 属性：指示页的区域性设置，该属性的值必须是有效的区域性 ID；

✧ Debug 属性：指示是否应使用调试符号编译该页。如果应使用调试符号编译该页，则为 true，否则为 false。由于此设置影响性能，因此只应在开发期间设置为 true；

✧ EnableEventValidation 属性：在回发和回调方案中启用事件验证。如果验证事件，则为 true，默认值为 true；

✧ EnableSessionState 属性：定义页的会话状态要求。如果启用了会话状态，则为 true；如果可以读取会话状态但不能进行更改，则为 ReadOnly；否则为 false，默认值为 true；

✧ EnableTheming 属性：指示是否在页上使用主题。如果使用主题，则为 true，默认值为 true；

✧ EnableViewState 属性：指示是否在页请求之间保持视图状态。如果要保持视图状态，则为 true，默认值为 true；

✧ ErrorPage 属性：定义在出现未处理页异常时用于重定向的目标 URL；

✧ Inherits 属性：定义供页继承的代码隐藏类，它可以是从 Page 类派生的任何类。此属性与 CodeFile 属性一起使用，后者包含指向代码隐藏类源文件的路径；

✧ Language 属性：指定在对页中的所有内联呈现（<% %> 和 <%= %>）和代码声明块进行编译时使用的语言；

✧ MasterPageFile 属性：设置内容页的母版页或嵌套母版页的路径；

✧ Title 属性：指定在响应的 HTML <title> 标记中呈现页的标题；

✧ Trace 属性：指示是否启用跟踪。如果启用了跟踪，则为 true，默认值为 false；

✧ UICulture 属性：指定用于页的用户界面 (UI) 区域性设置；

✧ ValidateRequest 属性：指示是否应发生请求验证。如果为 true，则请求验证将根据具有潜在危险值的硬编码列表检查所有输入数据；如果出现匹配情况，将引发 HttpRequestValidationException 异常，默认值为 true。

除了@ Page 指令外，ASP.NET 页框架还支持许多指令，每个指令都允许指定适合文件的不同选项。当使用指令时，虽然标准的做法是将指令包括在文件的开头，但是它们可以位于 .aspx 或 .ascx 文件中的任何位置。每个指令都可以包含一个或多个特定于该指令的属性。ASP.NET 页中常用的指令如表 1-6 所示。

表 1-6　ASP.NET 页中常用的指令

指　令	描　述
@ Page	定义 ASP.NET 页分析器和编译器使用的特定于页的属性。只能包含在 .aspx 文件中
@ Control	定义 ASP.NET 页分析器和编译器使用的控件特定属性。只能包含在 .ascx 文件（用户控件）中
@ Import	将命名空间显式导入页或用户控件中
@ Register	将别名与命名空间及类名关联起来，从而允许用户控件和自定义服务器控件在被包括到请求的页或用户控件时呈现
@ Assembly	在编译过程中将程序集连接到当前页，以使程序集的所有类和接口都可用在该页上
@ Master	标识 ASP.NET 母版页
@ PreviousPageType	提供用于获得上一页的强类型的方法，可通过 PreviousPage 属性访问上一页
@ MasterType	为 ASP.NET 页的 Master 属性分配类名，使得该页可以获取对母版页成员的强类型引用
@ OutputCache	以声明的方式控制页或用户控件的输出缓存策略
@ Reference	以声明的方式将页或用户控件连接到当前页或用户控件

4. 窗体标记

如果 ASP.NET 动态页面包含允许用户与页面交互并提交该页面的控件，则必须包含一个<form>标记，其属性定义了如何处理控件。虽然在页面上可以有许多个 HTML 窗体标记，但在.aspx 页中只能有一个服务器端窗体，可执行回发的服务器控件必须位于< form >标记之内。下面是一个典型的<form>标记：

```
<form id="form1" runat="server">
```

<form>标记属性包括：

✧ runat 属性：其属性值设置为 server；

✧ method 属性：默认值为 POST，但如果将 method 属性设置为 GET 或 POST 以外的其他值，则可能破坏内置视图状态和 ASP.NET 提供的回发服务；

✧ action 属性：始终设置为页本身的 URL，action 属性无法更改，因此，只能回发到页本身。

5. Page 类

Page 类与扩展名为 .aspx 的文件相关联，在运行时被编译为 Page 对象，并被缓存在服务器内存中，Page 类是一个用作 Web 应用程序的用户界面的容器控件。

利用 Page 对象的属性可以实现 Web 页面的一些信息，灵活使用 Page 属性可以实现动态设置页面信息，下面列出了常见属性及其用法。

✧ Page.Title 属性：获取或设置页的标题，如用户登录后在浏览器的标题栏中显示欢迎字样，Page.Title=txtusername.Text+“欢迎回来”；

✧ Page.Header 属性：在页声明中用 runat=server 定义了 head 元素的情况下获取或设置页的文档标头，还可以将样式表、样式规则、标题和源数据之类的信息添加到 head 元素中；

✧ Page.isPostback 属性：获取一个值，该值指示该页是否正为响应客户端回发而加载，或者它是否正被首次加载和访问，可以控制哪些代码只是在初次请求时执行；

✧ Page.ErrorPage 属性：获取或设置错误页，在发生未处理页异常的事件时请求浏览器将被重定向到该页，如 Page.ErrorPage = "ErrorPage.aspx"；

✧ Page.PreviousPage 属性：当使用 Transfer 方法或跨页发送在 ASP.NET 页之间传输处

理时，发送页中包含目标页中可能需要的请求信息，可以使用 PreviousPage 属性访问该信息，当用户直接从服务器请求该页时，PreviousPage 属性为 null 引用；

✧ Page.IsCrossPagePostBack 属性：指示跨页回发中是否涉及该页。ASP.NET 提供了两种用于在页与页之间传输控件的机制。可以使用 Transfer 方法在页之间进行传输处理，也可以通过将页 URL 指派给实现按钮控件的 PostBackUrl 属性来发出跨页请求。在任何一种情况下，PreviousPage 页属性都将包含表示上一页或发信方页的对象。例如，如果页 A 发送至页 B，则页 A 的 IsCrossPagePostBack 属性（可通过 PreviousPage 属性访问）将为 true，而页 B 的 PreviousPage 属性将具有页 A 的名称；

✧ Page.IsValid 属性：获取或设置页验证是否成功，如果页验证成功，则为 true；否则为 false；

✧ Page.Validators 属性：获取请求的页上包含的全部验证控件的集合。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 3

拓展任务号	1-3	任务名	样式管理工具的使用练习
计划用时	30 分钟	任务性质	课内
任务描述与目标			
通过样式可以很方便地控制网页的外观，目前用样式来布局网页已经是一种潮流，Visual Studio 2012 提供了可视化的样式操作工具，通过本练习熟练掌握使用这些工具			
主要操作步骤提示			
1．打开 Default.aspx； 2．选择一个 标记； 3．打开“新建样式”面板，分别选择不同的类别，进行样式设计； 4．通过管理样式对各个样式进行管理； 5．在页面中选择其他标记，在“运用样式”面板中进行选择			

1.4.2 创建事件处理程序

一、实战演练

（1）双击“Button”控件，页面自动切换到后台 Default.aspx.cs 代码编辑视图，可以看到开发工具自动在后台文件中添加了一个按钮事件。

（2）在该事件中添加的代码如下：

```
protected void Button1_Click(object sender, EventArgs e)
    {
        wel1.InnerHtml = "您好！" + TextBox1.Text + "，欢迎来到我的第一个网站";
    }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-7 所示。

表 1-7　演练完成情况评价表

任　务　号	1-5	任 务 名 称	给控件添加事件
任务子项	完成情况	主要问题	未完成原因
安装 IIS 信息服务组件			
新建一个虚拟目录			

三、知识点

1. Web 窗体事件

ASP.NET Web 服务器控件可以引发多种事件，许多事件是用户在浏览器中进行某些操作触发的。如在本项目中，浏览者通过单击按钮引发事件。

一般控件都有一个默认的事件，默认事件通过双击自动生成。如果想生成某个控件的非默认事件，则通过选择“属性”面板上的事件来生成。添加控件的非默认事件方法很简单，在窗体设计器中选择某个服务器控件，在“属性”面板上方单击按钮即可以选择添加事件。

事件生成后在.aspx 文件中相对应的控件标记与之进行关联，如下面的代码中“Button1”按钮有两个事件：Click 和 Load。

```
<asp:Button ID="Button1" runat="server" onclick="Button1_Click1" Text="确定"
onload="Button1_Load" />
```

事件过程用于在 Web 窗体上处理用户交互，它是动态交互式 Web 窗体对用户输入的典型反应。当一个用户 Web 窗体交互作用时就会产生一个事件，当事件产生的时候可以设计 Web 应用程序来执行适当的任务以处理这个事件。事件过程就是对产生的事件做出响应的行为。

Web 窗体包含两类事件，即客户端事件和服务器端事件。客户端事件过程和服务器端事件过程各有其优缺点。

在 Visual Studio 中编辑 ASP.NET 网页时，可以通过多种方法创建控件和页的服务器事件及客户端脚本事件。

1）客户端事件过程

客户端事件过程是在浏览网页的客户端上处理的事件。当事件产生时，不会给服务器发送信息而是由客户端浏览器解释代码执行相应的动作。客户端事件过程无权访问服务器端的资源，也不能访问 SQL Server 数据库。由于不需要与 Web 服务器之间的往返，客户端事件过程对于需要立即执行的事件是很有用的。

2）创建控件客户端事件处理程序

添加的控件客户端脚本事件一般采用像对 HTML 标记那样添加事件属性，当然也可以以编程的方式向 ASP.NET 控件添加客户端事件处理程序。

针对属性值添加要执行的客户端脚本，始终都要在属性中的客户端脚本之后添加一个分号(;)。这样，就可以保证控件的 AutoPostBack 属性设置为 true 的情况下首先运行此代码的客户端脚本。

代码清单 1-2 所示中演示了一个包含客户端脚本的 ASP.NET 网页，当用户将鼠标移到按钮上方时，此脚本就会更改该按钮的文本颜色。

代码清单 1-2　客户端脚本代码示例

```
<%@ Page Language="C#" %>
<html>
<head runat="server">
  <title>Untitled Page</title>
  <script type="text/javascript">
      var previousColor;
      function MakeRed()
      {
         previousColor = window.event.srcElement.style.color;
         window.event.srcElement.style.color = "#FF0000";
      }
      function RestoreColor()
      {
         window.event.srcElement.style.color = previousColor;
      }
  </script>
</head>
<body>
  <form id="form1" runat="server">
  <asp:button id="Button1" runat="server"
    text="Button1"
      onmouseover="MakeRed();"onmouseout="RestoreColor();" />
  </form>
</body>
</html>
```

3）服务器端事件过程

与客户端事件过程不同，服务器端事件过程需要把信息发送给 Web 服务器进行处理。尽管使用服务器端事件过程需要花费一定的时间，但是它比客户端事件过程具有更强的功能。

服务器端事件过程由位于 Web 服务器上已编译的代码组成，它既可以处理 Web 服务器控件产生的事件，也可以处理 HTML 服务器控件产生的事件，而且能够访问服务器资源。

由于服务器端事件过程需要在客户端与 Web 服务器之间往返，因此这些事件可能会影响网页的性能，所以它所支持的控件事件类型就受到限制，通常仅限于 Click 类型事件。一些服务器控件支持 Change 事件，如 CheckBox Web 服务器控件在用户单击该框时引发服务器代码中的 CheckedChanged 事件。一些服务器控件支持更抽象的事件，如 Calendar Web 服务器控件引发 SelectionChanged 事件，该事件为 Click 事件的更抽象版本。

服务器端事件过程支持单击事件和一个指定 Change 事件，但是它不支持一些频繁出现的事件，如鼠标键盘事件。但 Web 服务器控件仍然可以为这些事件调用客户端处理程序。

在服务器控件中，某些事件（通常是 Click 事件）会导致页被立即回发到服务器。

HTML 服务器控件和 Web 服务器控件（如 TextBox 控件）中的 Change 事件不会导致立即被发送，它们会在下一次发生发送操作时引发。但有时如 DropDownList、CheckBoxList 等一些选择类控件，一旦用户发生单击选择就需要该页被提交。如果将控件的 AutoPostBack 属性设置为 true，单击这类控件时会自动将该控件的状态发送到服务器进行处理。但 Button 控件不支持 AutoPostBack 属性设置。

有些服务器端控件同时支持客户端事件和服务器端事件。当一个控件中同时存在这两类事件，客户端事件会发生在服务器端事件之前，如下面的按钮不仅发生了客户端事件，同时也发生了服务器端事件。

```
    <asp:Button ID="Button1" runat="server" Text="Button" OnClientClick="show()"
onclick= "Button1_Click" />
```

服务器控件都有响应用户行为的一系列事件，这三种类型的服务器控件事件如下。

✧ 回发事件：促使 Web 页返回服务器并立即处理，从而影响执行的性能；

✧ 缓存事件：存储在页面的视图状态中，当出现在回发事件时被执行；

✧ 验证事件：在页面上被处理时不进行回发或缓存，验证控件支持这类事件。

4）创建控件服务器事件的处理程序

添加服务器事件处理程序，可以使用 Visual Web Developer 中的工具，也可以采用声明或代码方式。如果控件支持多个事件，通常将其中的一个事件配置为默认事件，如对于 Button 控件的默认事件是 Click 事件。创建服务器事件可以采用下面的方法。

✧ 在“设计”视图中，双击要为其创建默认事件处理程序的页面或控件。Visual Web Developer 用来创建默认事件的处理程序，并打开代码编辑器，此时插入点位于事件处理程序中；

✧ 如果要创建的是事件而不是控件的默认事件，则选择要创建其事件处理程序的控件，在“属性”面板中，单击事件符号  。在“事件名称”旁边的框中双击以便为该事件新建事件处理程序，设计器将使用“控件 ID_事件”约定为该处理程序命名；或在“事件名称”旁边的框中输入要创建的处理程序的名称；或在“事件名称”旁边的下拉列表中选择现有处理程序的名称。完成后，Visual Web Developer 将切换到代码编辑器，并将插入点置于处理程序中；

✧ 对于单文件的页面，可以在“源”视图窗口的顶部，从左侧的下拉列表中选择控件，然后从右侧的下拉列表中选择事件（已存在的事件以粗体显示）；

✧ 在 CS 代码中添加事件方法时，在 ASPX 文件的“源”视图中使用名为“On<事件名称>”的属性与事件方法程序挂钩。

删除事件处理程序，最快的方法是在删除了 CS 代码中添加事件方法后，在“源”视图中删除名为“On<事件名称>”的属性。如果多个控件使用同一个事件方法，则直接在“源”视图中删除要删除事件控件中名为“On<事件名称>”的属性。

5）为一个事件处理程序绑定多个事件

如果已具有一个事件处理程序，可以将几个控件事件绑定到该事件处理程序中。只要事件全部具有相同的方法签名，则这几个事件就可以来自同一个控件，而一个事件也可以来自多个不同的控件。

首先在 CS 代码中添加事件方法，在页标记中向每个控件添加相同的事件名称和方法名称，或在“事件”窗口中，在“事件名称”旁边的下拉列表中选择事件处理程序的名称。多个控件使用同一个事件的处理程序代码如下：

```
    <asp:Button ID="Button1" onclick="Button_Click" runat="server"  Text="
Button1" />
    <asp:Button ID="Button2" onclick="Button_Click" runat="server" Text="
Button2" />
    <asp:Button ID="Button3" onclick="Button_Click" runat="server" Text="
Button3" />
```

由于多个控件使用同一个事件处理程序，那么如何确定由哪个控件引发了事件呢？引发事件的控件会将控件对象传递给事件处理程序的第一个参数 object sender，可将这个参数强制转换为适当的控件类型。代码清单 1-3 演示了由几个不同按钮调用 Button 控件的 Click 事件处理程序，该处理程序显示引发事件按钮的 ID 属性。

代码清单 1-3 由几个不同按钮调用 Button 控件的 Click 事件处理程序代码

```
    private void Button_Click(object sender, System.EventArgs e)
    {
        Button btn;
        btn = (Button)sender;
        switch (btn.ID)
        {
            case "Button1":
                Label1.Text = "You clicked the first button";
                break;
            case "Button2":
                Label1.Text = "You clicked the second button";
                break;
            case "Button3":
                Label1.Text = "You clicked the third button";
                break;
        }
    }
```

6）服务器端事件参数

所有的服务器端事件都传递两个参数给事件处理过程，默认参数名称为 sender 和 e，即事件发送者和包含事件数据类的实例。其中后者通常是 EventArgs 类型，它不包含任何附加信息，然而，对于某些控件来说它是这些控件所特有的类型。如 ImageButton 控件为 Click 事件创建一个事件处理程序。在该事件处理方法中，事件参数对象的类型必须是 ImageClickEventArgs。在事件处理程序中，通过获取 ImageClickEventArgs 参数对象的 X 和 Y 属性，确定用户在图像上单击的位置。代码清单 1-4 演示了用户是否在规定的区域内触发 ImageButton 控件的 Click 事件。

代码清单 1-4 是否在规定的区域内触发 ImageButton 控件的 Click 事件代码。

```
    protected void ImageButton1_Click(object sender,  System.Web.UI.ImageClick
EventArgs e)
    {
        string msg = "";
        int x = e.X;
        int y = e.Y;
        if(x >= 50 && y >= 50)
        {
            msg = "Southeast";
        }
        else if(x >=50 && y < 50)
        {
             msg = "Northeast";
        }
        else if(x < 50 && y >= 50)
        {
            msg = "Southwest";
        }
        else if(x < 50 && y < 50)
        {
            msg = "Northwest";
        }
        Label1.Text = msg;
    }
```

2. Page 页生命周期事件

在 ASP.NET 页运行时，无论是页还是控件都会经历一个生命周期。在生命周期中将执行一系列处理步骤，每个处理步骤都会引发生命周期事件。如图 1-31 所示是 Page 页生命周期事件的顺序。

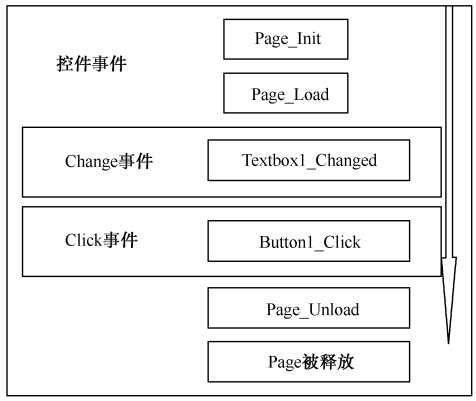

图 1-31　Page 页生命周期事件的顺序

在实际运用中可以使用 Page 事件来处理和维护 Web 页上的数据，对数据绑定做出响应及处理 Web 页上的异常。Page_Load 和 Page_Error 事件是在代码中最常用的事件。如代码清单 1-5 所示演示了页面导入时事件发生的顺序：

代码清单 1-5　页面导入时事件发生的顺序代码。

```
    protected void Page_Init(object sender, EventArgs e)
        {
            Response.Write("<script>alert('This is Init Event')</script>");
        }
        protected void Page_PreRender(object sender, EventArgs e)
        {
            Response.Write("<script>alert('This is PreRender Event')</script>");
        }
        protected void Page_Load(object sender, EventArgs e)
        {
            Response.Write("<script>alert('This is load Event')</script>");
        }
        protected void Page_LoadComplete(object sender, EventArgs e)
        {
            Response.Write("<script>alert('This is LoadCompleter Event')</
script>");
        }
```

Page 事件的 Load 事件通过双击页面的空白处后即可创建。在单文件页模型中，可以在“源”视图窗口的顶部，从左侧的下拉列表中选择 Page，然后从右侧的下拉列表中选择事件就可以创建相应的事件。在代码隐藏页模型中，要在 CS 文件中通过创建事件方法添加 Page 的其他事件。常见的 Page 事件如表 1-8 所示。

表 1-8　常见的 Page 事件

常见事件处理名称	发生时间说明
Page_DataBinding	当 Web 页面上服务器控件绑定到数据源时发生
Page_Error	当引发未处理的异常时发生
Page_Init	在 Web 页面的视图状态载入服务器控件并进行初始化时发生，是 Web 窗体生存期周期的第一步
Page_InitComplete	在页初始化完成时发生
Page_Load	当服务器控件加载到 Web 页面中时发生

续表

常见事件处理名称	发生时间说明
Page_LoadComplete	在页生命周期的加载阶段结束时发生
Page_PreLoad	在页 Load 事件之前发生
Page_PreRender	在加载页面之后、呈现之前发生
Page_Unload	当页面从内存中卸载时发生

3. Web 窗体逻辑代码

ASP.NET 使用三种方式为 Web 窗体添加逻辑代码。

1）混合代码

代码内容与 HTML 混合在一起，这种方式难于阅读和理解，混合代码要用<% %>括起来。如代码清单 1-6 所示演示了混合代码的使用方式。

代码清单 1-6 混合代码的使用方法。

```
<form id="form1" runat="server">
<div id="div122">
    <%
       int s = 0;
       for (int i = 0; i < Convert.ToInt32(TextBox1.Text); i++)
        {
                s = s + i;
         }
      %>
  </div>
  <p>s 的值为<%=s %></p>
</form>
```

2）单文件页模型

在单文件页模型中，页的标记及其编程代码位于同一个.aspx 文件中。如代码清单 1-7 所示演示了单文件页模型中 Button 控件的 Click 事件处理程序。

代码清单 1-7 单文件页模型代码。

```
<%@ Page Language="C#" %>
<script runat="server">
void Button1_Click(Object sender, EventArgs e)
{
   Label1.Text = "Hello world!";
}
</script>
<body>
  <form runat="server">
    <div>
     <asp:Label id="Label1"  runat="server" Text="Label"> </asp:Label> <br />
<asp:Button id="Button1" runat="server" onclick="Button1_Click" Text="Button">
     </asp:Button>
    </div>
  </form>
</body>
</html>
```

3）代码隐藏页模型

通过代码隐藏页模型，可以在一个文件（.aspx 文件）中保留标记，编程代码位于另一个

单独的文件中，这是 Visual Studio .NET 实现服务器代码的默认模型。代码隐藏页模型允许界面设计人员和代码开发人员同时进行工作。

在代码隐藏页模型中，不存在具有 runat="server" 属性的 script 块，而且@ Page 指令包含引用外部文件（如 regist.aspx.cs）和类的属性，这些属性将.aspx 页链接至其代码页。

说明：在代码隐藏页模型中使用 using 来导入命名空间，而在单文件页模型中使用指令来导入命名空间，如<%@ Import Namespace="System.Data.SqlClient"%>。

1.4.3 网站的调试与发布

第一个简易的网站创建完成后在发布之前，要进行生成和调试。Visual Studio 2012 自带了生成与调试的功能

一、实战演练

（1）执行“生成”→“生成网站”命令，如果在生成过程中发现网站存在错误，则会在开发环境的下方出现“错误列表”，把运行过程中的错误信息显示出来；如果没有问题，则会在 Visual Studio 开发环境的状态栏中出现“生成成功”。

（2）执行“调试”→“启动调试”命令或按【F5】键启动调试。如果是第一次启动调试，会弹出一个如图 1-32 所示的“未启用调试”对话框，单击“确定”按钮。

（3）创建 D:\ firstObjecFB 目录。发布网站时可以选择先发布到本地目录，再上传到指定的 Web 服务器。本例中选择先在本地发布的方式。如图 1-33 所示。

（4）执行“生成”→“发布网站”命令，打开“发布网站”对话框。在“目标位置”框中输入“D:\ firstObjecFB”，选择发布方式后单击“确定”按钮即可。

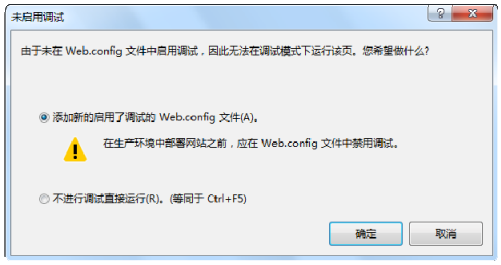

图 1-32 “未启用调试”对话框

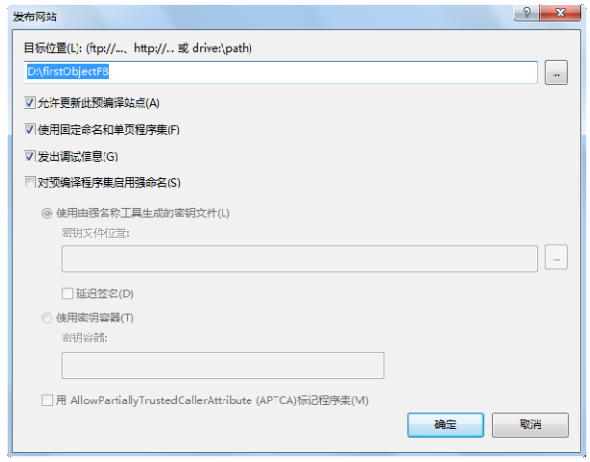

图 1-33 “发布网站”对话框

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 1-9 所示。

表 1-9 演练完成情况评价表

任 务 号	1-6	任 务 名 称	发布网站
任务子项	完成情况	主要问题	未完成原因
安装 IIS 信息服务组件			
新建一个虚拟目录			

三、知识点

Visual Web Developer 内置发布网站功能，将编译网站输出复制到指定的位置。当然也可以采用直接把整个网站目录复制到目标 Web 服务器的方式。同直接把整个网站目录复制到目标 Web 服务器方式相比，发布网站具有以下优点：

✧ 预编译过程中能发现任何编译错误，并在配置文件中标识；

✧ 单独页的初始响应速度更快，因为页已经过编译。如果不先编译页就将其复制到网站，则将在第一次请求时编译页，并缓存其编译输出；

✧ 将 App_Code 文件夹中的页、源代码等预编译到可执行输出中，并将可执行输出写入目标文件夹；

✧ 不会随网站部署任何程序代码，从而为代码文件提供了一项安全措施。

采用 Web Application 方式创建的网站发布方式略有不同，下面是 Web Application 项目的发布方式。

1）打开 VS2012，在菜单栏中选择“文件”→“新建”→“项目”，打开如图 1-34 所示的“新建项目”对话框。在对话框中选择“模板”→“Web” →“ASP.NET Web 窗体应用程序”。修改名称和位置，单击“确定”按钮。

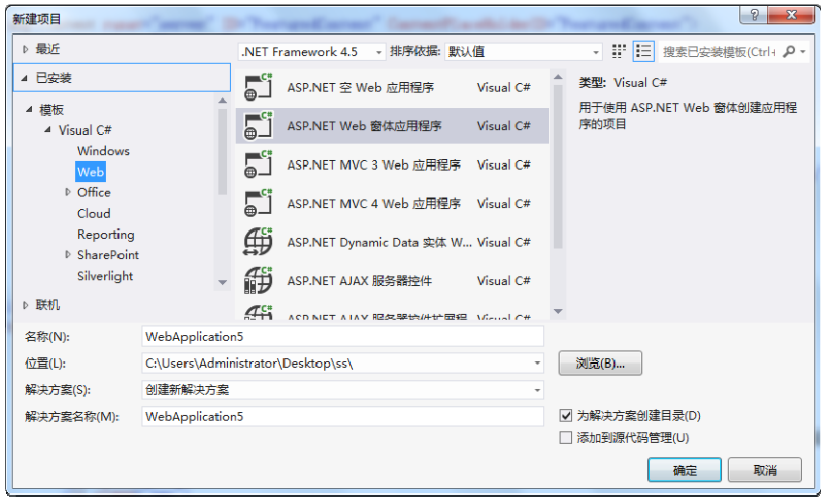

图 1-34 “新建项目”对话框

2）打开解决方案面板，右击项目名，在弹出的快捷菜单选择“生成”→“重新生成”，生成网站项目；再次右击项目名，选择“发布”，打开如图 1-35 所示的“发布 Web 应用程序”对话框。在菜单栏“生成”中可以找到相对应的功能选择。在对话框的“导入”中选择“新建”。在弹出的“新建配置文件”对话框中输入配置文件名。

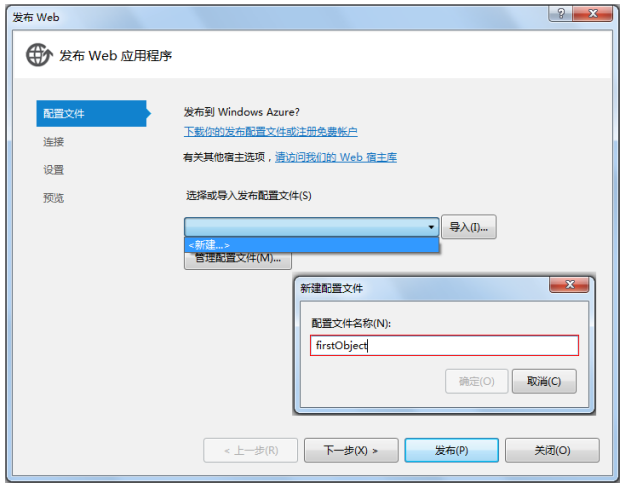

图 1-35　“发布 Web 应用程序”对话框

3）单击“下一步”按钮，在如图 1-36 所示的“发布方法”中选择“文件系统”，这样就可以发布到指定的本机文件夹上。

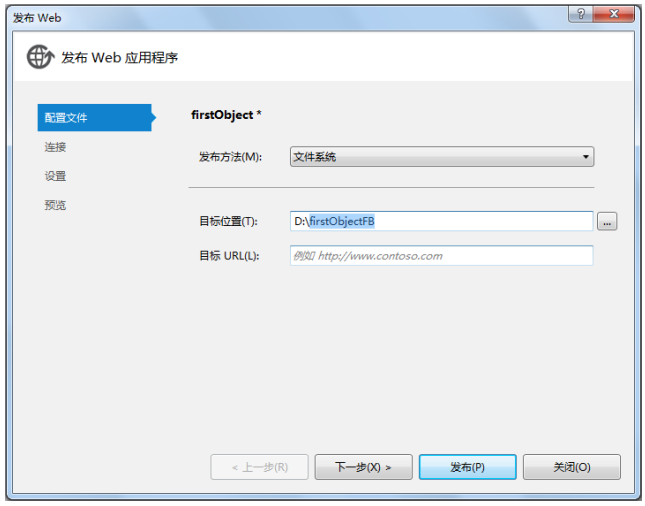

图 1-36　选择发布方法

课 外 练 习

拓展任务跟踪卡

拓展任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

<table>
<tr><td colspan="2">项目号</td><td></td><td>项目名</td><td></td></tr>
<tr><td colspan="2">项目完成方式</td><td colspan="3">□小组协作 □个人独立</td></tr>
<tr><td rowspan="9">项目完成情况（40%）</td><td rowspan="3">界面设计（20%）</td><td>自我评价</td><td colspan="2"></td></tr>
<tr><td>小组评价</td><td colspan="2"></td></tr>
<tr><td>个人评价</td><td colspan="2"></td></tr>
<tr><td rowspan="3">代码编写（20%）</td><td>自我评价</td><td colspan="2"></td></tr>
<tr><td>小组评价</td><td colspan="2"></td></tr>
<tr><td>个人评价</td><td colspan="2"></td></tr>
<tr><td rowspan="3">功能实现（10%）</td><td>自我评价</td><td colspan="2"></td></tr>
<tr><td>小组评价</td><td colspan="2"></td></tr>
<tr><td>个人评价</td><td colspan="2"></td></tr>
<tr><td rowspan="6">拓展与创新（30%）</td><td rowspan="3">创新能力（10%）</td><td>自我评价</td><td colspan="2"></td></tr>
<tr><td>小组评价</td><td colspan="2"></td></tr>
<tr><td>个人评价</td><td colspan="2"></td></tr>
<tr><td rowspan="3">设计潜力（10%）</td><td>自我评价</td><td colspan="2"></td></tr>
<tr><td>小组评价</td><td colspan="2"></td></tr>
<tr><td>个人评价</td><td colspan="2"></td></tr>
<tr><td colspan="5">主要存在问题</td></tr>
<tr><td colspan="5"></td></tr>
</table>

课外思考题

1．上网查看各类网站，了解打开网站首页与点开子页的地址栏中有哪些不同点？同时单击网站的超链接后，文件的后缀名都有哪些？上网查询这些后缀名都有什么含义？

2．上网对门户型网站、政府网站、企业网站、个人网站、娱乐网站、宣传网站及功能性的网站进行查看，说说这些不同种类网站的设计风格、色彩等方面有哪些相同和不同之处，各自有什么特点？

3．www.sina.com.cn 和 www.163.com 这两个网址中都有“com”，它们之间有什么不同？

4．目前网页的常用 CSS 样式表布局，请问 CSS 样式表作用有哪些？

5．请谈谈动态网页与静态网页在工作原理上有什么不同？

6．.NET 平台的核心技术是什么？在 Visual Studio .NET 中，解决方案资源管理器的作用是什么？

7．如何向项目中添加新 Web 窗体？

8．如果有一个带.aspx 扩展名的 Web 页，将如何验证它是否是一个 Web 窗体？又如何知道它是否具有 Web 服务器控件？Web 服务器控件在客户端产生什么类型的代码和脚本？

9．向 Web 窗体添加功能函数时使用代码隐藏页有何优点？怎样把一个代码隐藏页同一个.aspx 页链接在一起？

10．怎样链接一个事件过程与一个服务器控件的事件？设计一个事件过程，要用哪两个参数？

项目二

网络通讯录

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 掌握各种常用服务器控件及控件的常用属性、方法和事件；
- ✧ 掌握 Web 窗体的输入验证及 ASP.NET 验证控件的使用；
- ✧ 掌握创建用户控件的方法；
- ✧ 掌握内置对象的功能、属性、方法，了解其使用技巧；
- ✧ 了解 ADO.NET 的对象模型，掌握创建数据库的连接和对数据的操作方法。

技能目标

通过对本项目的学习，应该具备下面的能力：

- ✧ 能根据需要创建 Web 窗体，并能选择合理的 ASP.NET 服务器控件进行页面布局；
- ✧ 能选择合理的数据验证方式对输入数据进行验证；
- ✧ 能创建用户控件；
- ✧ 能使用 ADO.NET 的对象进行数据库的连接和数据操作；
- ✧ 能根据网络通讯录的功能分析创建完整的小型网站，对网站功能和代码进行调试并选择合理的方式发布网站。

教学建议

本项目计划总学时为 20 学时。

- ✧ 情境介绍：2 学时；
- ✧ 任务 1：8 学时；
- ✧ 任务 2：8 学时；
- ✧ 任务 3：2 学时。

教师在开始授课时，先给学生展示一个完整的项目，建立项目的完整理念。完成本项目的教学后，选择一些做得比较好的或有创新的项目进行展示，以此激励学生进行创新性学习。

在课内不可能完成全部项目内容，在整个教学过程中，许多内容需要学生利用课余时间去实现，如页面的美化、数据库的创建、功能相似的页面等。

2.1 情境介绍

谁都有因为手机的遗失而与许多好朋友失去联系的烦恼，为找回朋友的联系方式花费大量的时间和精力。采用纸质或一些电子形式的通讯录不仅有遗失之忧，同时携带不方便。为解决这些问题，可以开发一个网络通讯录，为用户提供方便。目前，人们不仅可以利用个人电脑，也可以利用手机上网，随时登录网站查找朋友的信息。

本项目的功能比较简单，设计界面也不复杂。项目比较贴近学生的生活，易于学生理解。本项目虽然比较简单，但包含了 ASP.NET 大部分的初步知识。

1. 需求分析

使用本网络通讯录，必须注册为网站用户，每一个用户都独立地拥有自己的通讯录。用户登录后可以对自己的联系人进行管理。如图 2-1 所示体现了通讯录用户体系中的一些重要环节。

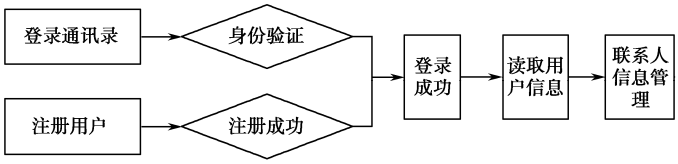

图 2-1　用户功能流程图

本通讯录中只有一个用户角色，即注册用户。未登录或未注册的浏览者只能看到通讯录使用说明等信息，只有经过注册的用户才能使用通讯录的管理功能。

2. 功能设计

（1）用户注册，浏览者在用户注册页面按照提示输入用户相关信息，注册功能中对注册用户进行检测，如果已经存在则提示注册失败。注册成功后转到联系人管理页面。

（2）用户登录，浏览者在登录页面输入用户名和密码，如果匹配成功则进入联系人的管理页面，如果不成功则提示登录失败信息。

（3）联系人管理功能，在这个页面中用户可以选择相关的功能进行操作，添加新联系人、查找所有联系人、按照类别查找联系人、删除联系人信息、修改联系人信息。

3. 数据库设计

根据通讯录的实际需求，创建 addressbook 数据库，建立一个如图 2-2 所示的注册用户信息表（userinfo）和一个如图 2-3 所示的联系人信息表（contactinfo）。

表 userinfo 的主键为 username，userid 设置为自动增量。

表 contactinfo 的主键为 conID，外键为 username，用于与 userinfo 表中的 username 字段关联。username 字段不可以为空。

4. 页面风格设计

网站不仅要实现特定的功能，还要反映设计者的审美情趣，以此来吸引访问者。网页设计则要表现出网站主题和应该实现的功能。

网站设计的实现可以分为两个部分。第一部分为站点的规划及草图的绘制，这一部分可以在纸上完成；第二部分为网页的制作。

目前网页内的文字、图像、动画等的页面布局排版，常用一些图片处理软件如 Photoshop 等来进行制作和设计，这部分往往由网页美工人员来完成。

列名	数据类型	允许空
userid	int	
username	varchar(20)	
userpassw	varchar(100)	
realname	varchar(50)	✓
sex	nchar(10)	✓
email	varchar(50)	
phone	char(20)	✓
headpicture	varchar(100)	✓
datetime	smalldatetime	✓
hobby	nvarchar(50)	✓
birthday	smalldatetime	✓
province	nchar(10)	✓

图 2-2　注册用户信息表（userinfo）

列名	数据类型	允许空
conID	int	
cname	nvarchar(20)	
sex	nchar(20)	✓
homephone	char(15)	✓
homeaddress	nvarchar(50)	✓
mobphone	char(15)	✓
company	nvarchar(50)	
comphone	char(15)	✓
comaddress	nvarchar(50)	✓
relation	nchar(10)	
username	varchar(20)	

图 2-3　联系人信息表（contactinfo）

接下来要做的就是通过软件将设计的蓝图变为网页，再由程序开发人员实现网站的功能。一个好的网站开发人员应该具备网站审美能力及简单的网页布局设计能力。

通讯录网站模拟人们现在常用的通讯录的方式设计。一个完整的通讯包括登录页面、注册页面、联系人信息管理页面，设计图如图 2-4 所示。

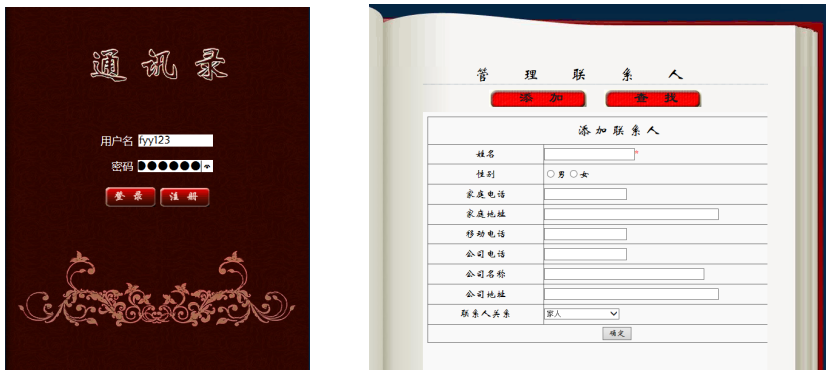

图 2-4　网络通讯录设计示图

5. 开发前准备

选择合适的位置新建一个网站文件夹“Address”。打开 Visual Studio 2012，执行“新建”→“网站”→“ASP.NET 空网站”命令，在“Web 位置”选择“文件系统”，选择刚刚创建的“Address”目录作为网站存放位置，创建“Address”网站。

在“解决方案面板”中右击“……Address”的项目名，在弹出的快捷菜单中选择“添加”→“新建文件夹”，创建一个“Images”文件夹用于存放网站图片。右击“Images”文件夹，在弹出的快捷菜单中选择“添加”→“现有项”，把准备好的图片添加到“Images”文件夹中。添加图片后如果看不到新添加的文件，可以单击“刷新”按钮，就可以看到所有新添加的文件。

2.2　任务 1　ASP.NET 服务器控件

任务描述

本任务主要通过创建网络通讯录完成各个页面的制作，学习 ASP.NET 各类服务器控件的应用。ASP.NET 服务器控件是 ASP.NET 的重要组成部分。几乎所有的页面都会包含一个或多个服务器控件。ASP.NET 服务器控件有各种各样的类型和大小，有简单控件和复杂控件。本任务主要学习 HTML 服务器控件、ASP.NET 标准控件和 ASP.NET 验证控件，同时还将学习如何

制作用户自定义控件。其它一些复杂的控件在项目 5 中介绍。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）能正确使用 HTML 服务器控件与 Web 服务器控件；

（2）掌握 HTML 在逻辑代码中的访问方式；

（3）掌握 ASP.NET 验证控件的使用方式及应用场合；

（4）能创建和应用用户自定义控件；

（5）完成网络通讯录的页面设计。

2.2.1 HTML 服务器控件

HTML 服务器控件是由 HTML 标记转换而来的，在默认情况下，ASP.NET 文件中的 HTML 元素作为文本进行处理，服务器无法使用 Web 窗体上的 HTML 元素。若要使这些元素能以编程方式进行访问，则需要添加 **runat="server"** 属性声明，将 HTML 元素作为服务器控件进行处理。还需要设置元素的 **id** 属性，才可以通过编程方式引用控件。

一、实战演练

（1）在 VS 中打开新建的“Address”网站添加用户注册页面。在“解决方案面板”中右击“……Address”的项目名，在弹出的菜单中选择“添加”→“添加新项”，打开“添加新项”对话框，选择“Visual C#” →“Web 窗体”，在对话框的下方把新建的文件名改为“regist.aspx”。在解决方案中添加“show”文件夹，右击“show”文件夹，在“添加新项”对话框中选择“样式表”，添加“style.css”样式文件。

（2）打开“regist.aspx”，单击“工作窗口”下方的“设计”，切换到“设计”视图。打开“管理样式面板”，单击“附加样式表”按钮，在弹出的如图 2-5 所示的“选择样式表”对话框中选择“show”文件夹中的“style.css”样式文件，单击“确定”按钮。

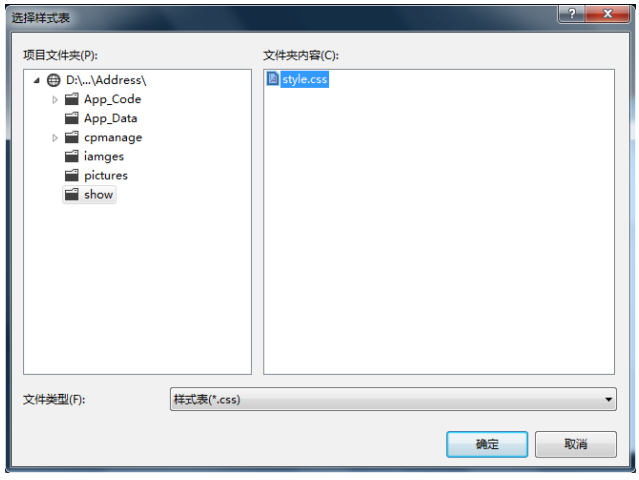

图 2-5 “选择样式表”对话框

（3）点击切换到“源”视图，可以看到在“regist.aspx”文件的“源代码”添加了样式附

加标记：

```
<link href="show/style.css" rel="stylesheet" type="text/css" />
```

在<title>中添加页面的标题，如下：

```
<title>网络通讯录注册表</title>
```

（4）回到“设计视图”选择“div”标记，在属性面板中添加其“ID”属性为“all”。在管理样式面板中单击“新建样式”按钮。在弹出的“新建样式”对话框中，“选择器”选择“#all”，具体设置如图 2-6 所示。也可以直接打开“style.css”样式文件输入样式标记。

（5）以同样的方法对“body”标记进行样式设置。设置后的 CSS 标记如下：

```
body
{
    text-align: left;
    background-color: #092A49;
    margin: 0px; text-align: left;
}
```

（6）打开工具箱，展开“HTML”选项卡，把 2 个“Div”拖到“#all”中，分别设置其 ID 号为“top”和“content”。在“content”中添加文字“通讯录”，在菜单栏中选择“格式”→“段落”，把“通讯录”三个字设置为<p>。

（7）从工具箱的“HTML”选项卡中拖动“table”到“content”，或在菜单栏中选择“表”→“插入表格”，在弹出的“插入表格”对话框中设置如图 2-7 所示的相关参数。在表格各行中输入注册用户信息所需的提示性文字。并对添加的内容进行样式设置。如代码清单 2-1 演示了添加样式和 HTML 控件后的“注册”页面。

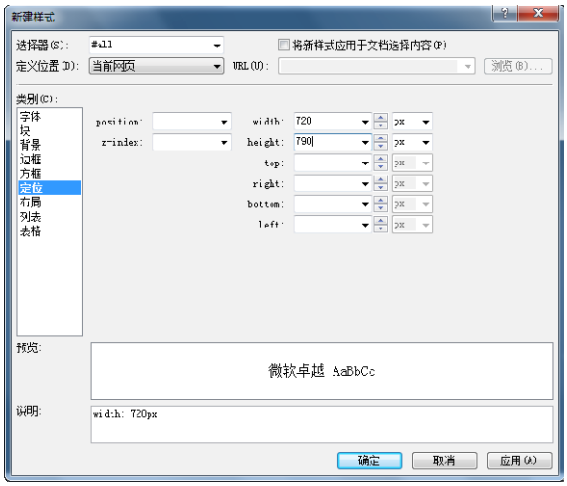

图 2-6 “新建样式”对话框

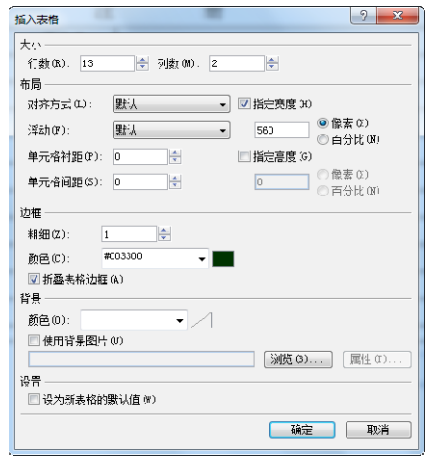

图 2-7 “插入表格”对话框

代码清单 2-1 注册页面的 HTML 代码。

```
<div id="all">
 <div id="top"> </div>
  <div id="content">
   <p>通讯录 </p>
    <table border="1" bordercolor="#000000" cellpadding="0px">
    <tr> <td id="title" colspan="2">注册</td> </tr>
      <tr>
```

```
        <td class="tdleft">用户名</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">密码</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">确认密码</td>
        <td class="tdright">  </td>
      </tr>
      <tr class="style6">
        <td class="tdleft">真实姓名</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">性别</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">邮箱</td>
        <td class="tdright">  </td>
      </tr>
      <tr>
        <td class="tdleft">电话</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">头像</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">爱好</td>
        <td class="tdright"></td>
      </tr>
      <tr>
        <td class="tdleft">生日</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td class="tdleft">通信地址</td>
        <td class="tdright"> </td>
      </tr>
      <tr>
        <td colspan="2"> </td>
      </tr>
    </table>
  </div>
  </div>
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-1 所示。

表 2-1 演练完成情况评价表

任务号	2-1	任务名称	HTML 控件—注册表页面设计
任务子项	完成情况	主要问题	未完成原因
使用 HTML 控件			
样式表的设计			
页面的布局			

三、知识点

添加了 **id** 属性和 **runat="server"** 属性声明后就可以通过编程方式引用控件，还可以通过设置属性 (Attribute) 来声明服务器控件实例上的属性 (Property) 参数和事件绑定。HTML 服务器控件运行在服务器上，并且直接映射到受大多数浏览器支持的标准 HTML 标记。

HTML 服务器控件在 System.Web.UI.HtmlControls 命名空间中。

1. HTML 服务器控件与 HTML 元素的映射关系

添加 HTML 服务器控件和添加任意 HTML 元素类似，可以在“设计”视图中通过工具箱将控件拖动到页面上，也可以在“源”视图中输入要用作控件元素的 HTML 语法。把 ID 属性设置为该页唯一的值，设置 runat="server"将元素转换为控件，可以在服务器代码中使用，如：

```
<div id="divotp"  runat="server"></div>
```

HTML 服务器控件与静态的 HTML 相关标记的使用方式相同。如在 HtmlImage 类对应 HTML<img> 元素。表 2-2 列出了常用 HTML 标记和 System. Web .UI.HtmlControls 命名空间中对应的 HTML 元素。

表 2-2 HTML 服务器控件与 HTML 元素的映射关系

序 号	HTML 服务器控件名	映射的 HTML 元素
1	HtmlAnchor	<a>
2	HtmlButton	<button>
3	HtmlForm	<form>
4	HtmlHead	<head>
5	HtmlImage	<img>
6	HtmlInputButton	<input type=button>
7	HtmlInputCheckBox	<input type=checkbox>
8	HtmlInputFile	<input type=file>
9	HtmlInputPassword	<input type="password"
10	HtmlInputRadioButton	<input type=radio>
11	HtmlInputText	<input type=text>
12	HtmlLink	<link>
13	HtmlMeta	<meta>
14	HtmlSelect	<select>
15	HtmlTable	<table>
16	HtmlTableCell	<td> 和 <th>
17	HtmlTableRow	<tr>
18	HtmlTextArea	<textarea>
19	HtmlTitle	<title>

2. HTML 服务器控件共享属性

作为 .NET Framework 的一部分，ASP.NET 共享命名空间和类之间的继承，有两个 HTML 控件的子集，它们共享不同基类的属性。这些子集称为容器控件和输入控件。

1）所有 HTML 控件共享的属性

在 HTML 控件上声明的任何属性都将添加到该控件的 Attributes 集合中，并且可以像属性那样以编程方式对它进行操作。例如，如果在 <body> 元素上声明 bgcolor 属性，即可以编程方式访问该属性并编写事件处理程序以更改它的值。HTML 控件共享的属性如下：

✧ Attributes。属性获取选定合同目录服务器控件标记上表示的所有属性名称/值对。

✧ Disabled。属性获取或设置一个值，该值指示 HTML 服务器控件是否被禁用。

✧ Style 属性。获取指定的 HTML 服务器控件的所有级联样式表 (CSS) 属性 (Property) 的集合。

✧ TagName 属性。可以用编程方式确定 HTML 服务器控件的元素名。例如，服务器端 元素的 TagName 属性 (Property) 包含“div”这个值。

2）HTML 输入控件共享的属性

HTML 输入控件映射到标准 HTML 输入元素。它们包含 Name 属性、Value 属性和 type 属性。

✧ Name 属性。获取或设置 HtmlInputControl 控件的唯一标识符名称。

✧ Value 属性。获取或设置与 HtmlInputControl 控件关联的值，等效于 HTML 元素的 value 属性。

✧ type 属性。获取 HtmlInputControl 的类型。

3. HTML 容器控件共享的属性

具有开始标记和结束标记的 HTML 服务器控件具有容器控件特性，使用 InnerText 和 InnerHtml 属性来获取或设置内容。

✧ InnerHtml 属性。获取或设置指定的 HTML 控件的开始和结束标记之间的内容，但不对 HTML 实体的特殊字符进行编码，如 Hello ，在浏览器中显示出粗体的“Hello”；

✧ InnerText 属性。获取或设置指定的 HTML 控件的开始和结束标记之间的所有文本。与 InnerHtml 属性不同，InnerText 属性会自动将特殊字符转换为 HTML 实体。对 HTML 实体的特殊字符进行编码，如 Hello ，在浏览器中显示的结果是“ Hello ”。

4. 以编程方式设置与读取 HTML 服务器控件属性

可以通过后台页面的代码根据需要动态设定控件属性，但是这些控件必须是服务器端控件。如：

```
myAnchor.HRef = "http://www.microsoft.com";        //HTML 服务器控件
```

以编程方式设置 HTML 服务器控件属性略有不同，除了可以进行一般的属性设置外，还可以设置 innerHTML 和 InnerText 属性。同时所有的 HTML 服务器控件还支持 Attributes 集合，该集合提供对所有控件属性的直接访问。如代码清单 2-2 所示演示了 HTML 服务器控件共享的属性使用。

代码清单 2-2 HTML 服务器控件共享的属性使用代码

HTML 标记清单：

```
<h3>HtmlControl 属性应用实例</h3>
<input id="Submit2" type="submit" value="Click Me!!" onserverclick="
Submit_Clicked"
```

```
            runat="server" style="color: red; background-color: Background;
font-size: 12px;" />
    <br />
    <span id="Message" runat="server"></span> <br />
    <h3>Submit2 Style 属性有</h3>
    <span id="Message1" runat="server"></span>
    <h3>Submit2 标记上的所有属性</h3>
    <span id="Message2" runat="server"></span>
```

CS 逻辑代码清单

```
    protected void Submit_Clicked(Object sender, EventArgs e)
        {
            Message.InnerHtml = "My TagName is: " + Submit2.TagName;
            IEnumerator keys = Submit2.Style.Keys.GetEnumerator();
            IEnumerator keys1 = Submit2.Attributes.Keys.GetEnumerator();
            while (keys.MoveNext())
            {
                String key = (String)keys.Current;
                Message1.InnerHtml += key + "=" + Submit2.Style[key] + "<br />";
            }
            while (keys1.MoveNext())
            {
                String key = (String)keys1.Current;
                Message2.InnerHtml += key + "=" + Submit2.Attributes[key] + "<br
/>";
            }
        }
```

5. 以编程方式添加 HTML 服务器控件

除了可以在页面中直接添加控件外，在很多时候 Web 页面中经常需要通过后台代码对控件进行读/写，或动态添加控件，也就是在页面的 CS 代码文件中通过编程方式进行添加。

如代码清单 2-3 所示演示了在页面中添加 HTML 服务器控件。

代码清单 2-3 在页面中添加 HTML 服务器控件代码

```
    protected void Button1_Click(object sender, EventArgs e)
        {
            HtmlTable tabl1 = new HtmlTable();
            tabl1.ID = "t1"; tabl1.Width = "100";
            tabl1.Height = "50"; tabl1.Border = 1;
            tabl1.BorderColor = "#0000ff";
            for (int rowcount = 0; rowcount < 5; rowcount++)
            {
               HtmlTableRow row = new HtmlTableRow();
                for (int cellcount = 0; cellcount < 4; cellcount++)
                {
                   HtmlTableCell cell;
                    if (rowcount <= 0)
                    {
                        cell = new HtmlTableCell("th");
    cell.InnerText=cellcount.ToString();              }
                    else
                    {
                        cell = new HtmlTableCell();
    cell.InnerText=rowcount.ToString()+","+cellcount.ToString(); }
                    row.Cells.Add(cell);
                }
                tabl1.Rows.Add(row);
```

```
        }
    //在<div>服务器控件中添加新生成的 table 表格。
            div1.Controls.Add(tabl1);
    }
```

在运行时创建控件比在设计时创建控件更可行，如有一个搜索结果页，要在其中以表的形式显示结果，由于不知道要返回多少项，可以将每个返回的项动态生成一个表。要通过编程向页添加控件，必须有放置新控件的容器，容器控件可以是 PlaceHolder 或 Panel Web 服务器控件，也可以是 、<p>等双标记的 HTML 控件。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 1

拓展任务号	2-1	任务名	完成添加通讯录网站联系人管理页面
计划用时	30 分钟	任务性质	课外
任务描述与目标			
用户登录后，就可以进行管理自己的联系人。联系人的管理包括：添加联系人，查找联系人，删除联系人，联系人的详细信息。这些功能可能通过一个管理页面来处理，并把这些文件统一放在一个文件夹中。			
主要操作步骤提示			

1．打开通讯录网站，右击项目名，在弹出的快捷菜单中选择“添加”→“新建文件夹”，把新建的文件夹命名为“cpmanage”。

2．在文件夹中添加文件 cpmanage.aspx，页面采用“<iframe id="iframe" name="if1" ></iframe>”标记，根据用户的选择显示页面，设计如图 2-8 所示。

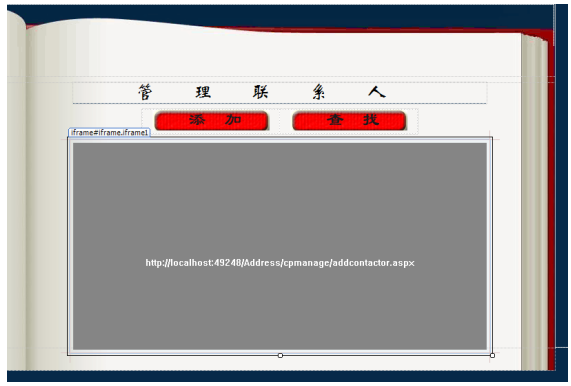

图 2-8　管理联系人页面

3．联系人管理页面的主要设计

4．参考代码如下。

```
<div id="all">
<div id="top"> </div>
<div id="content">
<p style="letter-spacing: 1em;" id="topic" runat="server">
 管理联系人</p>
<table id="menu">
<tr>
   <td> <img alt="" src="iamges1/bt1.jpg" /> </td>
   <td> <img alt="" src="iamges1/bt3.jpg" /> </td>
</tr>
</table>
<iframe id="iframe" name="if1" src="addcontactor.aspx" class="iframe1"></iframe>
</div>
</div>
</div>
```

拓展任务卡 2

拓展任务号	2-2	任务名	完成添加联系人页面设计
计划用时	30 分钟	任务性质	课外
任务描述与目标			
联系人页面与注册页面的布局相似，由管理页面引导入，因此设计更为简单			
主要操作步骤提示			
打开通讯录网站，右击“cpmanage”文件夹，在文件夹中添加文件“addcontactor.aspx”，页面设计如图 2-9 所示。			

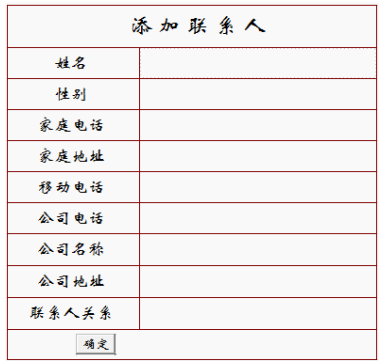

图 2-9 添加联系人页面设计

2.2.2 ASP.NET Web 标准服务器控件

Web 服务器控件与常见的 HTML 元素（如按钮和文本框）类似，但有些控件具有复杂行为，如日历控件和管理数据连接的控件。Web 服务器控件按照控件的功能分为标准控件、数据控件、验证控件、导航控件、登录控件、AJAX 服务器控件和 ASP.NET AJAX 扩展程序控件。本节将介绍的是工具箱中“标准”选项卡中的 ASP.NET Web 服务器控件——ASP.NET 标准服务器控件，其他类型的 Web 服务器控件将在其他章节中讲述。

一、实战演练

（1）在“解决方案面板”中打开“regist.aspx”。
（2）在“设计”视图中，从工具箱中拖入如图 2-11 所示的 Web 服务器控件。
（3）为每个控件设置属性，具体设置如表 2-3 所示。

表 2-3 注册页面 Web 服务器控件的属性设置表

控 件	ID	其 他 属 性
用户名文本框	txtName	默认
密码文本框	txtPassword	TextMode= Password
确认密码文本框	txtAffirm	TextMode= Password
真实姓名文本框	txtRname	默认
性别单选按钮表	radSex	RepeatDirection="Horizontal"
生日文本框	txtbirth	默认

续表

控　件	ID	其 他 属 性
邮箱文本框	txtEmail	默认
电话文本框	txtPhone	默认
通信地址文本框	txtAddress	默认
爱好复选按钮列表	chkHobby	RepeatDirection="Horizontal"
上传头像组件	filupImage	默认
确定注册按钮	btnCertain	Text="确定"
请选择按钮	btnSelect	Text="请选择"

（4）在“生日”栏附近添加一个 HTML 服务器控件 ，在这个 HTML 服务器控件中添加 Calendar 控件，然后单击该控件，单击 按钮，在“自动套用格式”对话框中选择“彩色型 2”。完成后的 regist.aspx 页面效果如图 2-10 所示。

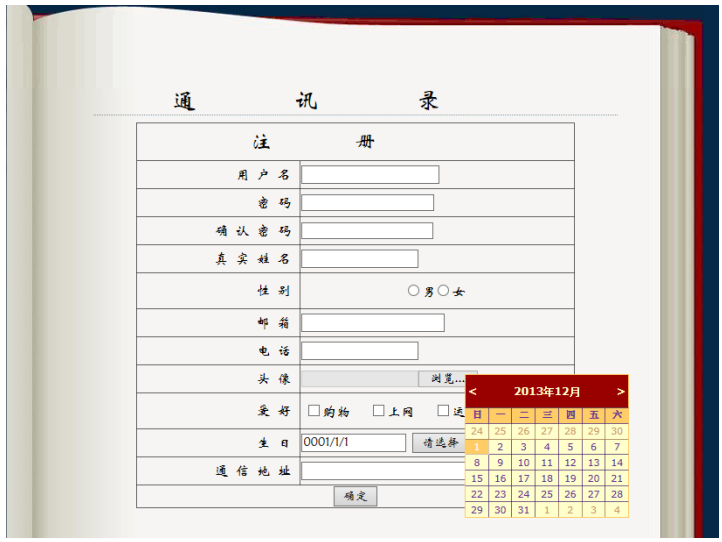

图 2-10　regist.aspx 页面的效果图

（5）双击页面空白处，添加“Page_Load”事件，在事件中添加如下代码：

```
caldiv.Visible = false;
txtbirth.DataBind();
```

（6）选择“Calendar”控件，在事件面板中双击“SelectionChanged”事件，在生成的“Calendar1_SelectionChanged”事件中添加如下代码：

```
string str1 = Calendar1.SelectedDate.ToString();
txtbirth.Text = str1.Substring(0, str1.IndexOf(" "));
caldiv.Visible = false;
```

（7）双击“btnSelect”控件控件，在“btnselect_Click”事件中添加如下代码：

```
caldiv.Visible = true;
```

（8）在工具箱中选择“按钮”查看浏览效果。并对控件的事件进行操作，理解事件的运行机制。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-4 所示。

表 2-4 演练完成情况评价表

任 务 号	2-2	任 务 名 称	ASP.NET 服务器控件
任务子项	完成情况	主要问题	未完成原因
控件的添加			
控件事件的添加			

三、知识点

1. 标准服务器控件格式

标准服务器控件都具有两个基本属性：ID 和 runat。runat="server"属性意味着这种控件逻辑运行在服务器上而不是用户的浏览器上。控件由“<asp:控件类名”开始。Web 服务器控件代码的基本格式如下：

```
<asp:TextBox ID="txtPhone" runat="server">
```

标准服务器控件是专门为 ASP.NET 创建的服务器控件。与 HTML 服务器控件不同，如果没有 runat="server"属性，Web 服务器控件就无法工作。同时 Web 服务器控件必须被放在<form id="form1" runat="server"> </form>中。

Web 服务器控件不仅包括窗体控件（如按钮和文本框），还包括一些特殊用途的控件。它与 HTML 服务器控件相比更为抽象，因为其对象模型不一定反映 HTML 语法。

向 ASP.NET 网页添加控件的方法与添加 HTML 元素的方法非常类似。可以使用可视设计器并从工具箱中添加控件，也可以通过在“源”视图输入控件的元素进行添加。

Web 服务器控件存在于 System.Web.UI.WebControls 命名空间内并可以被任何的 Web 窗体使用。

2. 服务器控件属性设置

每个服务器控件都有自己的属性，通过设置不同的属性，可以改变控件的展现内容和属性。

在“设计”视图中，选择控件，然后在“属性”窗口中设置属性；也可以在“源”视图中，在控件的元素标记中置入插入点，然后在“属性”窗口中设置属性。

通过“属性”窗口设置的属性会自动出现在该控件的 HTML 代码中。对一些比较熟悉的属性可以直接在 HTML 代码中编写。Visual Studio 2012 会根据控件的类型，给予自动提示。

3. 选择合适的控件

在创建 ASP.NET 页时，可以选择 Web 服务器控件，也可以选择 HTML 服务器控件，还可以在同一个页中混合使用这两种类型的控件。

在实际的应用中应尽量避免使用 HTML 服务器控件。Web 服务器控件比 HTML 服务器控件有更多的功能和更丰富的对象模型，包括大量 HTML 服务器控件中不存在的控件。但在页面设计时，能用静态的 HTML 标记就尽量不要用服务器控件。如显示页面文本，能用<SPAN>标记时就不要使用 Label 控件。

4. 常用的标准服务器控件简介

1）Label 控件

Label 控件是以动态方式在网页上显示文本的标签控件。通常当希望在运行时更改页面中的文本（如响应按钮单击）时使用。Label 控件使用的标准代码如下：

```
<asp:Label id="Label1"  Text="要在页面中显示的内容"  runat="server"/>
```

Label 控件的 Text 属性设置或读取在网页上显示的文本。如果要显示静态文本，可以使用 HTML 标记，而不需要使用 Label 控件。仅当需要以编程方式更改文本的内容或外观时，才使用 Label 控件。

Label 控件在浏览器中对应 HTML 控件是。

2）TextBox 控件

TextBox 控件是使用户可以输入文本的输入控件，配置不同的 TextMode 的属性，可以使 TextBox 有不同的显示模式。TextBox 控件对应<input type=text>HTML 控件。TextBox 控件使用的标准代码如下：

```
<asp:TextBox ID="TextBox1" TextMode="行为模式" Columns="文本框显示宽度"
MaxLength="最多允许的字符个数" ReadOnly="true | false" Rows="多行文本显示的行数"
Wrap=" true | false " CausesValidation=" true | false " runat="server"> Text 属性
值</asp:TextBox>
```

其常用属性如下。

✧ TextMode 属性：获取或设置 TextBox 控件显示行是单行、多行还是密码文本框。设置值为 TextBoxMode 枚举值之一。默认值为 SingleLine 表示为单行输入模式，MultiLine 表示为多行输入模式，Password 表示为密码输入模式；

✧ Text 属性：获取或设置 TextBox 控件中的文本，本属性在源代码中可以设置 Text 属性，也可以在开始和结束标记中设置，如<asp:TextBox ID="TextBox1" Text=" Text 属性值" runat="server"></asp:TextBox> 或 <asp:TextBox ID="TextBox1" runat="server"> Text 属性值 </asp:TextBox>。如果在源代码中两种方式都进行设置，那么 TextBox1.Text 的值则为后者；

✧ Columns 属性：获取或设置文本框的显示宽度（以字符为单位）；

✧ MaxLength 属性：获取或设置文本框中最多允许的字符数，该属性仅当 TextMode 属性设置为 TextBoxMode.SingleLine 或 TextBoxMode.Password 时才适用；

✧ Rows 属性：获取或设置多行文本框中显示的行数，默认值为 0，表示显示两行文本框。该属性仅当 TextMode 属性设置为 TextBoxMode.MultiLine 时才适用；

✧ Wrap 属性：获取或设置一个值，该值指示多行文本框内的文本内容是否换行。如果多行文本框内的文本内容换行，则为 true，否则为 false，默认值为 true。此属性仅当 TextMode 属性设置为 TextBoxMode.MultiLine 时才适用；

✧ ReadOnly 属性：获取或设置一个值，用于指示能否更改 TextBox 控件的内容，如果不能更改，则为 true，否则为 false，默认值为 false。将该属性设置为 true 时将禁止用户输入值或更改现有的值。注意，TextBox 控件的用户不能更改该属性，只有开发人员才可以更改；

✧ AutoPostBack 属性：获取或设置一个值，该值表示 TextBox 控件失去焦点时是否发生自动回发到服务器的操作。如果 TextBox 控件失去焦点时发生自动回发，则为 true，否则为 false，默认值为 false；

✧ CausesValidation 属性：获取或设置 TextBox 控件是否在回发发生时进行验证。当 TextBox 控件设置为在回发时进行验证，则值为 true，否则为 false，默认值为 false。在默认情况下，TextBox 控件失去焦点时不会导致页验证，若要将 TextBox 控件设置为在回发时进行验证，则要将 CausesValidation 属性和 AutoPostBack 属性设置为 true。

3）Button 控件

Button 控件主要使用户能够将页发送到服务器并触发页上的事件。它包括三种按钮控件，每种按钮控件在网页上显示的方式都不同。这三种按钮控件分别是 Button 控件、LinkButton 控件、ImageButton 控件。这三种控件使用的标准代码分别如下：

```
    <asp:Button ID="Button1" runat="server" Text="Button" />
    <asp:ImageButton ID="ImageButton1" runat="server" ImageUrl="图片地址" />
    <asp:LinkButton  ID="LinkButton1"  runat="server"> 连 接 显 示 文 本 </asp:Link
Button>
```

其常用属性如下。

✧ Text 属性：获取或设置控件上显示的文本，本属性对 Button 控件、LinkButton 控件有效，但也有差别，Button 控件没有单独的</asp:Button>结束标记，Text 属性只能在<asp:Button>内设置，而 LinkButton 控件的 Text 属性设置与 TextBox 控件相似；

✧ PostBackUrl 属性：获取或设置单击按钮控件时从当前页发送到的网页的 URL。如把 PostBackUrl 属性值设置为“default2.aspx”，则在单击按钮控件时将页发送到 default2.aspx，若不设置表示将页回发到自身。注意，如果设置属性的值为其他网站的地址，则不能正常显示该网站，如 PostBackUrl=http://www.baidu.com，则出现“您指定的网页无法访问”。该属性对三种控件都有效；

✧ ImageUrl 属性：获取或设置在 ImageButton 控件中显示图像的位置。该属性只对 ImageButton 控件有效；

✧ AlternateText 属性：获取或设置当图像不可用时，Image 控件中显示的替换文本。支持工具提示功能的浏览器将此文本显示为工具提示。该属性只对 ImageButton 控件有效；

✧ CommandName 属性：获取或设置命令名，该命令名与传递给 Command 事件的 Button 控件相关联，当在网页上具有多个 Button 控件时，可使用 CommandName 属性来指定或确定与每一个 Button 控件关联的命令名，可以用标识要执行的命令的任何字符串来设置 CommandName 属性，然后，可以以编程方式确定 Button 控件的命令名并执行相应的操作；

✧ CommandArgument 属性：指定补充 CommandName 属性的参数。尽管可以单独进行设置，但该属性通常只在设置了 CommandName 属性时才使用。

三种 Button 控件的使用方式基本一样。当单击这三种按钮控件中的任何一种时，都会向服务器提交一个窗体，使服务器处理网页中的逻辑功能代码，任何挂起的事件都会被触发。这些按钮还会引发自身的 Click 事件，从而可以在这些事件中编写处理程序。

Button 控件最主要的作用是用于页面数据的回送，在触发回送后，完成页面后台代码的处理逻辑，同时可以指定 Button 的 CommandName 属性来区分具体触发该事件的按钮。Button 控件最常用的事件是 Click 事件，如果不同种类的 Button 控件想使用同一段事件逻辑，则可以使用 Command 事件，用户的单击行为可以触发该事件。如果 Click 事件与 Command 事件同时存在，两个事件会同时被触发。下面的代码示例演示了如何使用 CommandName 属性来创建 Button 控件的 Command 事件。

在 ASPX 页面中添加三种 Button 控件的源代码如代码清单 2-4 所示，CS 文件的代码如代码清单 2-5 所示。

代码清单 2-4 三种 Button 控件的源代码。

```
    <asp:ImageButton ID="ImageButton1" runat="server" AlternateText="this is
IamgeButton" CommandName="images" ImageUrl="~/images/2.jpg" oncommand="Button1_
Click" />
    <asp:Button ID="Button1" runat="server" CommandName="enter" Text="确定"
oncommand= "Button1_Click" />
    <asp:LinkButton ID="LinkButton1" runat="server" CommandName="jump" oncommand="
Button1_Click"> 跳转</asp:LinkButton>
```

代码清单 2-5 CS 文件的代码。

```
    protected void Button1_Click(object sender, CommandEventArgs e)
    {
        switch (e.CommandName)
         {
           case "jump":
              Response.Redirect("http://www.163.com"); break;
           case "enter":
              Response.Write("<script language=javascript>alert('this is enter
Button');</script>");break;
           case "images":
              Response.Write("<script language=javascript>alert(' this is
IamgeButton '); </script>");break;
          }
        }
```

在后台逻辑中，根据按钮设定 CommandName 属性的不同将进行不同的处理。

4）CheckBox 和 CheckBoxList 控件

CheckBox 和 CheckBoxList 控件用于用户选择条件的复选框，使用单个的 CheckBox 控件与 CheckBoxList 控件相比，能更好地控制页面上的布局。如果想要用数据源中的数据创建一系列复选框，则 CheckBoxList 控件是更好的选择。CheckBox 和 CheckBoxList 控件使用的标准代码分别如下：

```
    <asp:CheckBox ID="CheckBox1" Checked="true|false" runat="server" />
    <asp:CheckBoxList ID="CheckBoxList1" DataSourceID ="绑定的数据源控件的 ID 号"
DataTextField="显示文本对应的数据源字段" DataValueField="选择项值对应的数据源字段"
runat="server">
    <asp:ListItem Selected="true|false" Value="object1" ></asp:ListItem>   //
选择项
    </asp:CheckBoxList>
```

其常用属性如下：

✧ Checked 属性：获取或设置一个值，该值指示是否已选中该 CheckBox 控件，该属性对 CheckBox 控件有效。如果选定此项，则为 true，否则为 false，默认值为 false；

✧ Selected 属性：获取或设置是否选定此项，如果选定此项，则为 true，否则为 false，默认值为 false；

✧ Text 属性：获取或设置列表控件中为 ListItem 所表示的项显示的文本，设置方式与 TextBox 控件的 Text 属性设置相同；

✧ Value 属性：获取或设置与 ListItem 关联的值。注意，如果 Text 属性和 Value 属性只设置其中一个，则这一个属性起到两个属性的作用，如果两个属性都设置了，则分别起作用；

✧ SelectedItem 属性：获取 CheckBoxList 列表控件中索引最小的选定项，这是一个只读属性。

如果 CheckBoxList 和数据源的某个字段相关联，则 DataSourceID 属性是设置或获取数据源控件的 ID 号，DataTextField 和 DataValueField 属性所对应的字段值相当于 ListItem 的 Text 属性和 Value 属性；如果不使用绑定数据源控件 ID 号的方式，则可以通过在 CS 代码中设置 DataSource 属性的方式绑定数据源。

如何选择多个 CheckBox 作为一组复选框使用，如代码清单 2-6 和代码清单 2-7 所示分别演示了如何判断 CheckBox 和 CheckBoxList 列表处于选择状态。

代码清单 2-6 CheckBox 选择状态判断代码。

```
 protected void Button1_Click1(object sender, EventArgs e)
{
CheckBox[] checks = new CheckBox[] { CheckBox1, CheckBox2, CheckBox3 };
string checkResult="您选择的结果为：";
  foreach(CheckBox check1 in checks)
     {
       if (check1.Checked)
          checkResult = checkResult + check1.Text + ",";
     }
//去除生成字符串的最后一个字符逗号
checkResult= checkResult.Remove(checkResult.Length-1);
Response.Write("<script  language=javascript>alert('"  +  checkResult  +
"');</script>");
}
```

代码清单 2-7 CheckBoxList 列表选择状态判断代码。

```
protected void Button1_Click1(object sender, EventArgs e)
{
    string checkResult="您选择的结果为：";
       for (int i = 0; i < CheckBoxList1.Items.Count; i++)
       {
          if(CheckBoxList1.Items[i].Selected)
             checkResult = checkResult +CheckBoxList1.Items[i].Text + ",";
       }
    checkResult=checkResult.Remove (checkResult.Length-1);
}
```

5）RadioButton 和 RadioButtonList 控件

RadioButton 和 RadioButtonList 控件允许用户选择列表的其中一项。RadioButton 控件很少单独使用，往往通过 GroupName 属性将同一组的 RadioButton 控件分配到一个组中，在一个组内，每次只能选择一个单选按钮。组名称可以任意，具有相同组名称的所有单选按钮将视为单个组的组成部分，RadioButtonList 控件中的列表项将自动进行分组。RadioButton 和 RadioButtonList 控件使用的标准代码分别如下：

```
    <asp:RadioButton  ID="RadioButton1"  runat="server"  Checked="true | false"
GroupName="r1" />
    <asp:RadioButtonList ID="RadioButtonList1"  runat="server">
       <asp:ListItem Selected="true | false"> </asp:ListItem>
    </asp:RadioButtonList>
```

RadioButton 和 RadioButtonList 控件基本用法与 CheckBox 和 CheckBoxList 控件非常相似，不再详细说明。

6）ListBox 和 DropDownList 控件

ListBox 和 DropDownList 控件都能够使用户从预定义的列表中选择项，不同的是 ListBox 控件可以一次显示多个项并能够使用户选择多个项，而 DropDownList 控件只能显示和选择一项，选择项列表通过下拉按钮显示。ListBox 和 DropDownList 控件使用的标准代码分别如下：

```
<asp:ListBox ID="ListBox1" runat="server" SelectionMode="Multiple|Single" >
    <asp:ListItem>hhhh</asp:ListItem>
</asp:ListBox>
<asp:DropDownList ID="DropDownList1" runat="server">
    <asp:ListItem Selected="True">111</asp:ListItem>
</asp:DropDownList>
```

所选项可在 ListBox 和 DropDownList 控件的 SelectedItem 属性中得到。如果将 ListBox 控件设置为单选模式，则此属性将返回一个选定的项；如果设置为多选模式，则通过循环遍历整个 Items 集合并检查每个项的 Selected 属性来获取选定的项。

7）Image 控件

Image 控件主要用于在 ASP.NET 网页上显示图像，并用代码管理这些图像，可以在设计时或运行时以编程方式为 Image 对象指定图形文件，还可以将控件的 ImageUrl 属性绑定到一个数据源，根据数据库信息显示图形。Image 控件使用的标准代码如下：

```
<asp:Image ID="Image1" runat="server" AlternateText="替代文本" DescriptionUrl="描述文本地址" ImageAlign="图片对齐方式" ImageUrl="图片地址" BorderColor="边框颜色" BorderStyle="边框样式" BorderWidth="边框宽度" />
```

其常用属性如下：

✧ AlternateText 属性：获取或设置当图像不可用时，Image 控件中显示的替换文本；

✧ DescriptionUrl 属性：获取或设置图像详细说明的位置。DescriptionUrl 属性指定提供图像附加详细信息的单独页面，这就意味着把浏览器链接到另外的页面，这可能会造成理解上的困难。另外浏览器对于本属性的支持也是不一致的，并且不是非常完善。如果说明信息量比较大，则可以采用链接的方式；

✧ ImageUrl 属性：获取或设置在 Image 控件中显示图像的位置。图像的 URL 分为相对 URL 或绝对 URL；

✧ ImageAlign 属性：获取或设置 Image 控件相对于网页上其他元素的对齐方式。ImageAlign 属性值通过 ImageAlign 枚举值来完成设置，ImageAlign 枚举值如表 2-5 所示。

表 2-5　ImageAlign 枚举值

成员名称	说　明
NotSet	未设定对齐方式
Left	图像沿网页的左边缘对齐，文字在图像右边换行
Right	图像沿网页的右边缘对齐，文字在图像左边换行
Baseline	图像的下边缘与第一行文本的下边缘对齐
Top	图像的上边缘与同一行上最高元素的上边缘对齐
Middle	图像的中间与第一行文本的下边缘对齐
Bottom	图像的下边缘与第一行文本的下边缘对齐
AbsBottom	图像的下边缘与同一行中最大元素的下边缘对齐
AbsMiddle	图像的中间与同一行中最大元素的中间对齐
TextTop	图像的上边缘与同一行上中最高文本的上边缘对齐

与其他大多数 Web 服务器控件不同，Image 控件不支持任何事件。如 Image 控件不响应鼠标单击事件。如果在页面上静态地显示图片，应用 HTML 的<img>标记。

8）ImageMap 控件

ImageMap 控件用于创建具有用户单击的单个区域的图像，这些单个区域称为作用点，每一个作用点都可以是一个单独的超链接或回发事件。ImageMap 控件由两个元素组成，第一个是图像，它可以是任何标准 Web 图形格式的图形；第二个是 HotSpot 控件的集合。每个作用点控件都是一个类型为 CircleHotSpot、RectangleHotSpot 或 PolygonHotSpot 的不同项。对于每个作用点控件，都要定义用于指定该作用点位置和大小的坐标。ImageMap 控件使用的标准代码如下：

```
    <asp:Imagemap id="Vote" Imageurl="图片地址" Width="图宽" Height="图高"
Alternatetext="替代文本"
    HotsPotMode=" NotSet｜Navigate｜PostBack｜Inactive" runat="Server">
    <asp:RectangleHotSpot top="0" left="0" bottom="200" right="200"
PostBackValue="Yes"
    Alternatetext="Vote yes" Target="目标窗口或框架名"></asp:RectangleHotSpot>
    <asp:CircleHotSpot NavigateUrl="~/Default2.aspx" X="300" Y="100"
    Radius="100" PostBackValue="No" Alternatetext="Vote no" />
    <asp: PolygonHotSpot coordinates="128,185,335,157,400,224,400,400,228,400"
    PostBackValue="sohu" Alternatetext="Southern Region"> </asp:PolygonHotSpot>
    </asp:imagemap>
```

其常用属性如下：

✧ HotsPotMode 属性：获取或设置单击 HotSpot 对象时 ImageMap 控件的 HotSpot 对象的默认行为；

✧ NavigateUrl 属性：获取或设置单击 HotSpot 对象时导航至的 URL；

✧ Target 属性：获取或设置目标窗口或框架，单击导航至 URL 的 HotSpot 对象时在其中显示链接到的网页内容；

✧ PostBackValue 属性：可以为 HotSpot 对象指定名称。单击 HotSpot 以后，此名称将在 ImageMapEventArgs 事件的数据中传递。仅当 HotSpotMode 属性设置为 HotSpotMode.PostBack 时，此属性才对生成到服务器的回发有效。

CircleHotSpot 的作用点范围由 X、Y 和 Radius 三个属性值来决定，其中 X、Y 是圆心坐标，Radius 为圆的半径。RectangleHotSpot 的作用点范围通过设置由左上角坐标点（top、left）和右下角坐标点（bottom、right）来确定。PolygonHotSpot 的作用点范围由 coordinates 属性设置，coordinates 属性值是各个点的坐标。

9）FileUpload 控件

FileUpload 控件允许用户将文件从用户的计算机发送到服务器上的指定位置，用户能够传送图片、文本文件或其他文件，同时可实现限制传送文件的大小和检查文件类型。FileUpload 控件使用的标准代码如下：

```
    <asp:FileUpload id="FileUpload1" runat="server"> </asp:FileUpload>
```

其常用属性如下：

✧ FileBytes 属性：从 FileUpload 控件指定的文件中获取一个字节数组；

✧ FileContent 属性：获取 Stream 对象，它指向要使用 FileUpload 控件上载的文件；

✧ FileName 属性：获取客户端上使用 FileUpload 控件上载文件的名称，FileName 属性返

回的文件名而不包含此文件在客户端上的路径；

✧ PostedFile 属性：获取使用 FileUpload 控件上载文件的基础 HttpPostedFile 对象，还可访问上载文件的其他属性。可以使用 ContentLength 属性来获取文件的长度，使用 ContentType 属性来获取文件的 MIME 内容类型。此外，还可以使用 PostedFile 属性来访问 FileName 属性、InputStream 属性和 SaveAs 方法；

✧ HasFile 属性：获取一个值，该值指示 FileUpload 控件是否包含要上载的文件。在对要上载的文件执行操作之前，应使用该属性来验证该文件是否存在。

HasFile 属性的常用方法如下：

SaveAs()方法：将上载文件的内容保存到 Web 服务器上的指定路径。

FileUpload 控件由一个文本框控件和一个浏览按钮组成，浏览按钮可以使用户选择客户端上的文件并将它上载到 Web 服务器。用户通过在控件的文本框中输入本地计算机上的文件完整路径来指定要上载的文件，也可以通过单击“浏览”按钮，然后在“选择文件”对话框中定位文件来选择。

用户选择要上载的文件后，FileUpload 控件不会自动将该文件保存到服务器，而必须通过一个事件机制来保存指定文件，可以调用 SaveAs()方法将文件内容保存到服务器上的指定路径。在调用 SaveAs()方法将文件保存到服务器之前，应使用 HasFile 属性来验证 FileUpload 控件是否包含文件，若 HasFile 返回 true，则调用 SaveAs()方法；如果它返回 false，则向用户显示消息。调用 SaveAs()方法时，必须指定用来保存上载文件目录的完整路径。如果没有在应用程序代码中显式指定路径，则当用户试图上载文件时会引发异常。该行为可防止用户在应用程序目录结构的任意位置进行写操作及访问敏感的根目录，有助于确保服务器上文件的安全。

使用 SaveAs()方法将上载的文件写到指定的目录，要求 ASP.NET 应用程序必须具有服务器上该目录的写访问权限。

10）AdRotator 控件

AdRotator 控件提供一种在 ASP.NET 网页上显示广告图形的方法。当用户单击广告时，系统会将他们重定向到指定的目标 URL。该控件会从使用数据源（通常是 XML 文件或数据库表）提供的广告列表中自动读取广告信息，如图形文件名和目标 URL。

AdRotator 控件显示 XML 文件中的广告和显示数据库中的广告所设置的属性是不一样的。显示 XML 文件中的广告只要将 AdRotator 控件的 AdvertisementFile 属性设置为 XML 文件的路径即可，将广告的信息存储在 XML 文件中，但是用于显示广告的 XML 说明文件必须遵守规定的格式，XML 文件包括下列可选属性：

✧ ImageUrl 属性：要显示图像的 URL；

✧ NavigateUrl 属性：单击 AdRotator 控件时要转到的网页的 URL；

✧ AlternateText 属性：图像不可用时显示的文本；

✧ Keyword 属性：用于筛选特定广告的广告类别；

✧ Impressions 属性：指示广告可能显示频率的数值（加权数值）。在 XML 文件中，所有 Impressions 值的总和不能超过 2 048 000 000 – 1；

✧ Height 属性：广告的高度（以像素为单位）。此值会重写 AdRotator 控件的默认高度设置；

✧ Width 属性：广告的宽度（以像素为单位）。此值会重写 AdRotator 控件的默认宽度设置。

为更好地保证安全，可以将 XML 文件扩展名命名为.xml 之外的名称，如 .adr 等，同时把这个 XML 放在 App_Data 文件夹中。

如代码清单 2-8 所示是 XML 文件显示广告的示例应用。

代码清单 2-8 XML 文件显示广告代码

```
    <asp:AdRotator ID="AdRotator1" AdvertisementFile="~/App_Data/XmlAdR.adr"
runat="server" />
    XmlAdR.adr 文件代码：
    <?xml version="1.0" encoding="utf-8" ?>
    <Advertisements
xmlns="http://schemas.microsoft.com/AspNet/AdRotator-Schedule-File">
        <Ad>
          <ImageUrl>~/Images/2010419145925.jpg</ImageUrl>
          <height>60</height>
          <width>190</width>
          <NavigateUrl>http://www.163.com</NavigateUrl>
          <AlternateText>网易</AlternateText>
          <Impressions>80</Impressions>
          <Keyword>Topic1</Keyword>
        </Ad>
        <Ad>
          <ImageUrl>~/Images/2010419145930.jpg</ImageUrl>
          <height>90</height>
          <width>90</width>
          <NavigateUrl>http://www.sohu.com</NavigateUrl>
          <AlternateText>搜狐</AlternateText>
          <Impressions>80</Impressions>
          <Keyword>Topic2</Keyword>
        </Ad>
    </Advertisements>
```

说明：在 Advertisements 元素中，为每个要包括在广告列表中的广告创建一个 Ad 元素。

如果把广告放在数据库中，那么 AdRotator 控件中要对下面的属性进行设置：

✧ AlternateTextField 属性：获取或设置一个数据字段，该字段中包含替代文本；

✧ ImageUrlField 属性：获取或设置一个数据字段，该字段应该包含图片的地址，相当于设置 ImageUrl；

✧ NavigateUrlField 属性：获取或设置一个数据字段，该字段应该包含广告的连接地址，相当于设置 NavigateUrl。

常用的数据库存放广告方式代码如下：

```
    <asp:AdRotator ID="AdRotator1" runat="server" AlternateTextField=" imagename"
DataSourceID="SqlDataSource1" ImageUrlField=" imageaddress" Navigate UrlField="
interaddress"  Width="200" Height="100" />
```

11）Calendar 控件

Calendar 控件用于显示日历中的可选日期，并显示与特定日期关联的数据。用户通过此日历可移动到任意一年中的任意一天，并一次显示一个月份中的日期，同时共显示六周。Calendar 控件使用的标准代码如下：

```
    <asp:Calendar ID="Calendar1" runat="server"  DayNameFormat="Full" First
DayOfWeek="Monday" ShowGridLines="True">
```

其常用属性如下：

✧ DayNameFormat 属性：指定 Calendar 控件上星期的显示格式。使用 DayNameFormat 的枚举值之一来设置该属性，可以指定一周中各天是以全名显示、以简称（缩写名称）显示、以该日的首字母显示还是以该日的前两个字母显示；

✧ FirstDayOfWeek 属性：获取或设置要在 Calendar 控件的第一天列中显示的一周中的某天。用 FirstDayOfWeek 的枚举值之一来设置该属性, FirstDayOfWeek 的枚举值有 Sunday、Monday、Tuesday、Wednesday、Thursday、Friday、Saturday 和 Default；

✧ ShowGridLines 属性：获取或设置一个值，该值指示是否用网格线分隔 Calendar 控件上的日期；

✧ SelectedDate 属性：获取或设置确定选定的日期。

Calendar 控件的常用事件是 SelectionChanged 事件，当用户在单击日期选择器控件选择一天、一周或整月时触发该事件。

Calendar 控件可显示个别日的特定详细信息，如任务列表、事件时间表或类似的信息。Calendar 控件采用.NET 的 DateTime 对象，可以显示 0～9999 的任何日期。用户可通过单击一个日期或在不同的月份之间移动以移动到任意日期，还可将日历配置为允许用户选择多个日期，包括整周或整月。Calendar 控件可以通过样式设置来修改外观。

12）BulletedList 控件

BulletedList 控件用于创建一个无序或有序的项列表，它们分别映射为 HTML 的<ul>或<ol>标记。BulletedList 控件使用的标准代码如下:

```
    <asp:BulletedList ID="BulletedList1" runat="server" BulletStyle="Custom
Image" Width="136px">
      <asp:ListItem>列表项显示文本</asp:ListItem>
      <asp:ListItem Selected="True">列表项显示文本</asp:ListItem>
    </asp:BulletedList>
```

其常用属性如下：

✧ BulletStyle 属性：获取或设置 BulletedList 控件的项目符号样式。通过 BulletStyle 枚举来指定项目符号样式，该样式可应用于 BulletedList 控件中的列表项。BulletStyle 枚举值如表 2-6 所示；

✧ BulletImageUrl 属性：指定 BulletedList 控件中显示每个项目符号图像的路径。若要为项目符号指定自定义图像，还必须将 BulletStyle 属性设置为 CustomImage；

✧ FirstBulletNumber 属性：指定排序 BulletedList 控件中开始列表项编号的值。如果 BulletStyle 属性设置为 Disc、Square、Circle 或 CustomImage 字段，则忽略分配给 FirstBulletNumber 属性的值。

表 2-6 BulletStyle 枚举值

成员名称	说明
NotSet	不设置项目符号样式而由浏览器决定要显示的项目符号样式
Numbered	项目符号样式为数字（1、2、3…）
LowerAlpha	项目符号样式为小写字母（a、b、c…）
UpperAlpha	项目符号样式为大写字母（A、B、C…）
LowerRoman	项目符号样式为小写罗马数字（i、ii、iii…）
UpperRoman	项目符号样式为大写罗马数字（I、II、III…）
Disc	项目符号样式为实心圆
Circle	项目符号样式为空心圆
Square	项目符号样式为实心四方形
CustomImage	项目符号样式为自定义图像

BulletedList 控件与 ListBox、DropDownList 和其他 ASP.NET 列表控件派生自相同的 ListControl 类，因此，使用 BulletedList 控件就像使用那些控件一样。可以通过创建静态项或将控件绑定到数据源来定义 BulletedList 控件的列表项。如果在设计时知道要显示哪些项，可在标记中将控件的 Items 集合设置为一组单独的项；如果要显示的项是动态的，可以通过代码在运行时创建项集合。

13）HyperLink 控件

HyperLink 控件可在网页上创建链接，可以使用户在应用程序中的页间移动。HyperLink 控件映射为 HTML 的<a>标记，HyperLink 控件使用的标准代码如下：

```
    <asp:HyperLink id="hyperlink2" ImageUrl="图片地址" NavigateUrl="链接到的地址
"  Target="目标窗口或框架名"  runat="server">  </asp:HyperLink>
```

其常用属性如下：

✧ ImageUrl 属性：获取或设置为 HyperLink 控件显示图像的路径。如果同时设置了 Text 和 ImageUrl 属性，则 ImageUrl 属性优先，如果图像不可用，则显示 Text 属性中的文本。在支持工具提示功能的浏览器中，Text 属性也变成提示工具；

✧ NavigateUrl 属性：获取或设置单击 HyperLink 控件时链接到的 URL。

使用 HyperLink 控件的主要优点是可以在服务器代码中设置链接属性。如可以根据页面中的条件动态更改链接文本或目标页。使用 HyperLink 控件的另一个优点是，可以使用数据绑定来指定链接的目标 URL，目标 URL 指向用户可以在其中读取有关产品更多详细信息的页面。

14）Panel 控件

Panel 控件在网页中提供了一种容器控件，可以用作静态文本和其他控件的容器，在 HTML 中映射为 标记。Panel 控件使用的标准代码如下：

```
    <asp:Panel id="Panel1" runat="server" BackColor="gainsboro" Height="200px"
Width="300px">显示内容</asp:Panel>
```

Panel 控件在编程方式生成控件、隐藏/显示一组控件或本地化一组控件时，尤其有用。如代码清单 2-9 所示演示了编程方式生成控件。

代码清单 2-9 编程方式生成控件代码

```
    void Page_Load(Object sender, EventArgs e)
    {
       for (int i=1; i<=3; i++)
      {
          Label l = new Label();
          l.Text = "Label" + (i).ToString();
          l.ID = "Label" + (i).ToString();
          Panel1.Controls.Add(l);
          Panel1.Controls.Add(new LiteralControl("<hr/>"));
          TextBox t = new TextBox();
          t.Text = "TextBox" + (i).ToString();
          t.ID = "TextBox" + (i).ToString();
          Panel1.Controls.Add(t);
          Panel1.Controls.Add(new LiteralControl("<br />"));
      }
    }
```

15）PlaceHolder 控件

PlaceHolder 控件主要用于动态地将子元素添加到容器中，在运行时动态添加、删除或依次通过子元素。该控件只呈现其子元素而不呈现自身的任何标记。

16）MultiView 和 View 控件

MultiView 和 View 控件是一对组合使用的控件，主要用于其他控件或标记的容器，并提供了一种方便地显示信息时替换显示页面内容的方式。MultiView 控件用于一个或多个 View 控件的容器，View 控件又可以包含标记和控件的任何组合。MultiView 控件一次显示一个 View 控件，并显示该 View 控件中的所有标记和控件。MultiView 和 View 控件使用的标准代码如下：

```
<asp:MultiView ID="MultiView2" ActiveViewIndex="0" runat="server">
    <asp:View ID="View1" runat="server"> </asp:View>
    <asp:View ID="View2" runat="server"> </asp:View>
</asp:MultiView>
```

其常用属性如下：

✧ ActiveViewIndex 属性：获取或设置 MultiView 控件的活动 View 控件的索引。View 控件的索引是由它在 MultiView 控件中的声明顺序确定的，声明第一个 View 控件的索引为 0。

如代码清单 2-10 所示演示了通过单击数字轮动图片的显示，显示结果如图 2-11 所示。

图 2-11　单击数字后的不同显示结果

代码清单 2-10　通过单击数字轮动图片的显示代码

```
    <asp:Panel  ID="Panel1"  runat="server"  CssClass="dq"  Height="205px"  Width="
180px">
      <asp:MultiView ID="MultiView1" ActiveViewIndex="0" runat="server">
         <asp:View ID="View1" runat="server">
          <asp:Image ID="Image1" runat="server" ImageUrl="~/images/2010419145958.
jpg" />
         </asp:View>
         <asp:View ID="View2" runat="server">
           <asp:Image  ID="Image2" runat="server"  ImageUrl="~/images/2010419145937.
jpg" />
         </asp:View>
         <asp:View ID="View3" runat="server">
           <asp:Image ID="Image3" runat="server" ImageUrl="~/images/2010419145925.
jpg" />
         </asp:View>
       </asp:MultiView>
       <asp:LinkButton  ID="LinkButton1"  runat="server"  onclick="LinkButton1_
Click"> 1</asp:LinkButton>
       <asp:LinkButton  ID="LinkButton2"  runat="server"  onclick="LinkButton1_
Click"> 2</asp:LinkButton>
       <asp:LinkButton  ID="LinkButton3"  runat="server"  onclick="LinkButton1_
Click"> 3</asp:LinkButton>
    </asp:Panel>
    事件代码：
    protected void LinkButton1_Click(object sender, EventArgs e)
        {
```

```
        LinkButton lb=(LinkButton)sender;
        MultiView1.ActiveViewIndex =Convert.ToInt32(lb.Text)- 1;
    }
```

17）Wizard 控件

Wizard 控件可以把网页的内容分成多步骤呈现，使得网页可以使用多个步骤来完成不同类型的数据输入。控件的每个步骤都具有一个 StepType 属性，用于确定该步骤所具有的导航功能的类型。Wizard 控件使用的标准代码如下：

```
    <asp:Wizard ID="Wizard2"   runat="server">
      <WizardSteps>
        <asp:WizardStep runat="server" StepType="Start" title="Step 1"> </asp:
WizardStep>
         <asp:WizardStep runat="server" title="Step 2">     </asp:WizardStep>
      </WizardSteps>
    </asp:Wizard>
```

Wizard 控件分为两部分，元素<asp:Wizard>用于设置导航文本，每一步的具体内容显示在<asp:WizardStep> 中 ； 元 素 <asp:Wizard> 通 过 对 StartNextButton 、 StepPreviousButton FinishCompleteButton、NavigationButton 等各类属性设置自定义样式的<asp:Wizard>控件。

如代码清单 2-11 所示简单地演示了使用 Wizard 控件分步完成数据的输入，运行效果如图 2-12 所示。

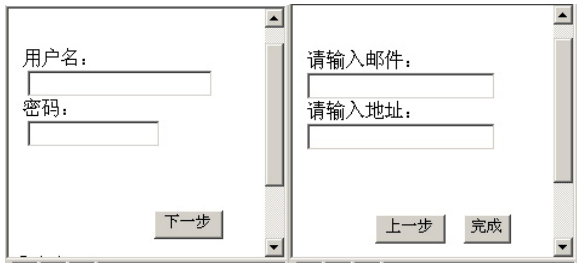

图 2-12　Wizard 控件分步的运行效果

代码清单 2-11　使用 Wizard 控件分步完成数据的输入代码

```
    <asp:Wizard  ID="Wizard1"  runat="server"   ActiveStepIndex="0"  Height="167px"
Width="180px"  onfinishbuttonclick="Wizard1_FinishButtonClick" >
    <WizardSteps>
      <asp:WizardStep runat="server"  StepType="Start">
    用户名：<br /><asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
    密码：<br /><asp:TextBox ID="TextBox2"  runat="server" TextMode="Password" >
</asp:TextBox>
      </asp:WizardStep>
      <asp:WizardStep runat="server" >
        请输入邮件：<br /><asp:TextBox ID="TextBox3" runat="server"></asp: TextBox>
        请输入地址：<br /><asp:TextBox ID="TextBox4" runat="server"></asp: TextBox>
      </asp:WizardStep>
    </WizardSteps>
    </asp:Wizard>
```

18）Table、TableRow 和 TableCell 控件

Table 控件主要用于网页上使用代码进行编程的表。TableRow 和 TableCell 控件提供显示 Table 控件内容的方法。如果创建的是静态表则应使用 HTML 表而不要使用 Table 控件。Table、TableRow 和 TableCell 控件使用的标准代码分别如下：

```
<asp:Table ID="Table1" runat="server">
    <asp:TableRow runat="server" >
        <asp:TableCell runat="server">表格中显示文本</asp:TableCell>
    </asp:TableRow>
</asp:Table>
```

可以通过设计方式为 Table 控件来添加行列，在 Table 控件的“属性”面板中单击“rows”属性，打开如图 2-13 所示的“TableRow 集合编辑器”对话框，通过“添加”和“移除”按钮添加和删除行。选择新添加的行，在“属性”项中选择“Cells”属性，打开如图 2-14 所示的“TableCell 集合编辑器”对话框，可以添加和删除表的单元格。

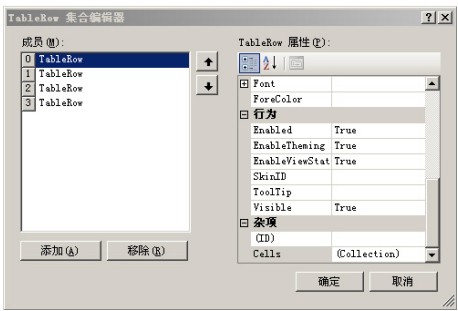

图 2-13 “TableRow 集合编辑器”对话框

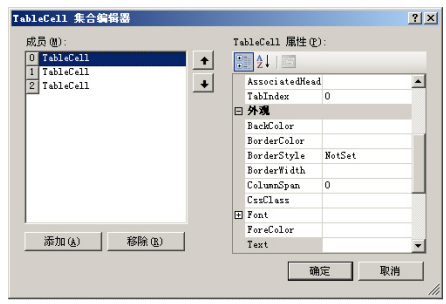

图 2-14 “TableCell 集合编辑器”对话框

如果将表中要显示的内容添加到 TableCell 控件中，可以将单元格的 Text 属性设置为任意 HTML 文本，也可以通过将控件添加到单元格的 Controls 集合，在单元格中显示控件。

Table 控件比 HtmlTable 控件更易于编程，Table 控件可作为 TableRow 控件的父控件。表中支持一个名为 Rows 的属性，该属性是 TableRow 对象的集合，通过添加或删除此集合中的项，指定表中的行。TableRow 控件又支持名为 Cells 的集合，该集合包含 TableCell 对象。

19）HiddenField 控件

HiddenField 控件用于存储非显示值的隐藏字段，它呈现为 <input type= "hidden"/> 元素。HiddenField 控件使用的标准代码如下：

```
<asp:HiddenField ID="HiddenField1" runat="server" Value="存储非显示值" />
```

通常情况下，Web 窗体页的状态由视图状态、会话状态和 Cookie 来维持。但是，如果这些方法被禁用或不可用，则可以使用 HiddenField 控件来存储状态值。

20）Literal 控件

Literal 控件作为页面上其他内容的容器。Literal 控件使用的标准代码如下：

```
<asp:Literal ID="Literal1"  runat="server">h</asp:Literal>
```

对于静态内容，无需使用容器，可以将标记作为 HTML 直接添加到页面中。Literal 控件不向文本中添加任何 HTML 元素，因此不支持包括位置属性在内的任何样式属性。在实际中一般不会使用 Literal 控件来添加动态网页内容。

5. *列表控件的通用属性与方法*

大多数列表控件都有一些共同的特点，即具有多个项。这就涉及项目的添加、删除和修改，同时也就存在了选择项这一属性。

如果要对列表项控件添加列表项，可以通过如图 2-15 所示的“ListItem 集合编辑器”对话框进行添加。此对话框可以通过单击控件“属性”面板中“Items”属性后面的...按钮打开；

也可通过单击列表项控件后的按钮，拉开列表项控件的任务菜单，选择“编辑项”打开。在对话框中可以通过“添加”和“移除”按钮进行增减项目，并通过对话框右边的“列表项属性”来设置属性，单击↑和↓按钮可以调整各列表项的顺序。

图 2-15 “ListItem 集合编辑器”对话框

大多数列表项控件都有 SelectedIndexChanged 事件，列表控件的选定项在信息发往服务器之间变化时触发该事件。

在大多数列表项控件中都可以通过一个 Items 属性获取列表项控件的集合 ListItemCollection。这个集合可以通过索引下标访问列表项，同时也可以对这个集合进行添加、删除、清空列表项操作。如表 2-7 所示列出了 Items 属性的常用方法及作用。

表 2-7 Items 属性的常用方法及作用

方 法 名	作 用
Add()	将列表项追加到集合的结尾
Clear()	从集合中移除所有列表项对象
Insert()	将列表项插入集合中的指定索引位置
Remove()	从集合中移除列表项
RemoveAt()	从集合中移除指定索引位置的列表项
Contains()	确定集合是否包含指定的项
FindByText()	搜索集合中具有 Text 属性且包含指定文本的列表项
FindByValue()	搜索集合中具有 Value 属性且包含指定值的列表项

四、任务拓展

本节完成两个课外拓展实践任务。

拓展任务卡 3

拓展任务号	2-3	任务名称	Web 服务器控件的使用练习
计划用时	30 分钟	任务性质	课外
任务描述与目标			
在注册表页面的设计中用到了一部分 Web 服务器控件，但有很多服务器控件并没有使用到，通过课外扩展练习熟练使用这些比较复杂的服务器控件，为后面的学习打下良好的基础			
主要操作步骤提示			
1．新建一个网站 Execise，用于控件的练习； 2．为每个控件添加一个页面，如练习 AdRotator 控件，文件名为 AdRotator_Exercise.apsx； 3．要求熟练完成 Table 系列控件、Wizard 控件、MultiView 和 View 控件、BulletedList 控件、AdRotator 控件、ImageMap 控件、FileUpload 控件的属性设置，了解常用事件			

拓展任务卡 4

拓展任务号	2-4	任务名	完成添加通讯录网站的其他页面
计划用时	30 分钟	任务性质	课外

任务描述与目标

在完成注册表页面后，比较容易就能设计出添加联系人页面。由于添加联系人页面会加载在其他页面中，所以页面设计相对比较简单

主要操作步骤提示

1．打开通讯录网站，打开添加联系人页面“addcontactor.aspx”，为页面添加相关控件，参考页面设计如图 2-16 所示；

2．添加联系人查找页面“fingcp.aspx”，页面的参考设计如图 2-17 所示。设计的要点是在页面的左边添加 标记，右边添加 Web 服务器控件中的 Panel 控件，Panel 控件源 HTML 代码设置为：

```
<asp:Panel ID="Panel1" CssClass="newStyle1"
runat="server"></asp:Panel>
```

注意：左边采用静态的 标记主要是为了提高页面的访问速度，右边使用 Panel 控件主要是为了满足动态添加控件的需要。

3．在页面的左边添加六个“LinkButton”按钮，把这六个按钮的 Text 属性根据联系人的类别分别进行设置；

4．添加一个 DropDownList 控件，主要用于选择查找条件。控件设置后的代码如下：

图 2-16　添加联系人页面

```
<asp:DropDownList ID="DropDownList1" runat="server">
    <asp:ListItem>姓名</asp:ListItem>
    <asp:ListItem Value="家庭地址">家庭地址</asp:ListItem>
    <asp:ListItem>公司地址</asp:ListItem>
</asp:DropDownList>
```

5．添加用于关键字查找的 TextBox 控件和一个完成查找逻辑代码的按钮控件 Button。控件设置后的代码如下：

```
<asp:TextBox ID="TextBox1" runat="server" Width="116px"></asp:TextBox>
<asp:Button ID="Button1" runat="server" Text="查找" onclick="Button1_Click" />
```

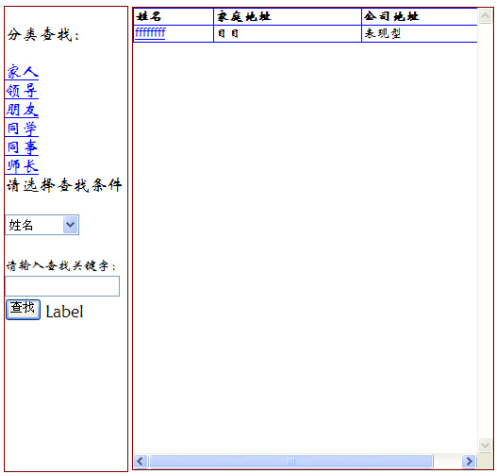

图 2-17　联系人查找页面

图 2-18　显示详细信息页面

6．添加显示详细信息页面 detailinfo.aspx，参考页面设计如图 2-18 所示。设计的要点是联系人详细信息与修改联系人信息分别在“Panel1”和“Panel2”两个容器控件中。第一次访问时的 Panel1.Visible = true、Panel2.Visible = false，两个 Panel 控件的定位位置相同。通过“显示信息”和“修改信息”两个按钮实现显示转换

2.2.3 ASP.NET 服务器验证控件

用户在输入信息时，错误的输入会延误用户的输入工作，甚至可能中断 Web 应用程序。输入验证是检验 Web 窗体中用户的输入是否和期望的数据值、范围或格式相匹配的过程。输入验证可以减少错误信息的等待时间及返回错误的可能性，也可以减少由于用户的输入问题而导致的 Web 站点崩溃的可能性，这都将改善用户访问 Web 站点的体验。将输入验证控件与易理解和有用的错误信息相结合，可以极大地提高 Web 应用程序的可行性，从而提高客户对网站整体质量的印象。

一、实战演练

为注册页面 regist.aspx 添加输入验证功能

（1）打开 regist.aspx 页面的“设计”视图，在“用户名”、“密码”、“确认密码”及“邮箱”文本框后添加“*”，标记为必填文本。

（2）在工具箱中打开验证选项卡，把 RequiredFieldValidator 分别拖到设置“*”的文本框后面。

（3）选择用户名后面的 RequiredFieldValidator 控件，打开“属性”面板，对 ControlToValidate 和 ErrorMessage 这两个属性进行设置。设置后的控件代码如下：

```
    <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
ControlToValidate= "txtName" ErrorMessage="请输入用户名"Display="None"SetFocusOn
Error="True"CssClass="yz"> </asp:RequiredFieldValidator>
```

对另外几个必填文本框添加并绑定 RequiredFieldValidator 控件，其他属性设置相同。

（4）在“确认密码”文本框后面添加 CompareValidator 控件，在“属性”面板中设置相应的属性。设置后的控件代码如下：

```
    <asp:CompareValidator ID="CompareValidator1" runat="server" ControlTo
Compare="txtPassword" ControlToValidate="txtAffirm" CssClass="yz" Error
Message="密码输入不相同">
    </asp:CompareValidator>
```

（5）在“邮箱”文本框后添加 RegularExpressionValidator 控件，打开“属性”面板，选择 ValidationExpression 属性，单击“选择”按钮，打开如图 2-19 所示的“正则表达式编辑器”对话框。在“标准表达式”中选择“Internet 电子邮件地址”。设置后的控件代码如下：

```
    <asp:RegularExpressionValidator ID="Regular
ExpressionValidator1" runat=" server" Control
ToValidate= "txtEmail" CssClass="yz" ErrorMessage="
请输入正确的邮箱格式" ValidationExpression= "\w+([-+.']\
w+)*@\ w+([-.]\w+)*\.\w+([-.]\w+)*">
    </asp:RegularExpressionValidator>
```

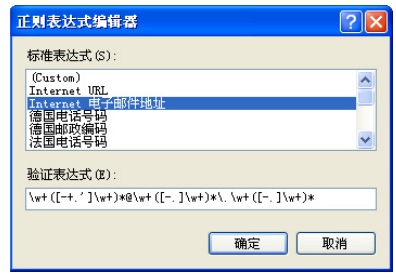

图 2-19 “正则表达式编辑器”对话框

（6）在“电话号码”文本框中添加 RegularExpressionValidator 控件，打开“属性”面板，选择 ValidationExpression 属性，在“正则表达式编辑器”对话框的“标准表达式”中选择“中华人民共和国电话号码”。设置后的控件代码如下：

```
    <asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="
server" ControlToValidate= "txtPhone" CssClass="yz" ErrorMessage="电话号码格式不
正确" ValidationExpression="(\(\d{3}\)|\d{3}-)?\d{8}">
    </asp:RegularExpressionValidator>
```

（7）在“用户名”项 RequiredFieldValidator 控件后面添加 RegularExpressionValidator 控件，打开“属性”面板，选择 ValidationExpression 属性，在“正则表达式编辑器”对话框的“标准表达式”中选择“(Custom)”，在“验证表达式”中输入“^[A-Za-z0-9_]{6,18}$”，要求用户在注册时只能输入 6～18 位的数字、字母和下画线。设置后的控件代码如下：

```
    <asp:RegularExpressionValidator ID="RegularExpressionValidator3" runat="
server" ControlToValidate= txtName" CssClass="yz" ErrorMessage="6～18 个字符可以
字母数字和下画线" ValidationExpression= ^[A-Za-z0-9_]{6,18}$"></asp:Regular
ExpressionValidator>。
```

同样在“密码”项后面添加 RegularExpressionValidator 控件，在“验证表达式”中输入“^[A-Za-z0-9_]{6,18}$”，允许用户输入 6～18 位的数字、字母和下画线。

（8）在“用户名”项 RequiredFieldValidator 控件后面添加 CustomValidator 控件，这是一个用户自定义控件，用于检测用户名的唯一性。设置后的控件代码如下：

```
    <asp:CustomValidator ID="CustomValidator1" runat="server" ontrolTo
Validate="txtName" CssClass="yz" ErrorMessage="用户名已存在" SetFocusOn
Error="True" > </asp:CustomValidator>。
```

（9）在注册表的下面添加一个层，把这个层移到表格下方合适的位置。向层中添加 ValidationSummary 控件，使用默认属性。

（10）设置后的注册页面效果如图 2-20 所示。

图 2-20　设置后的注册页面效果

（11）运行页面，查看设置效果。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-8 所示。

表 2-8 演练完成情况评价表

任务号	2-6	任务名称	添加注册表验证
任务子项	完成情况	主要问题	未完成原因
添加验证控件			
验证控件的属性设置			
验证信息布局			

三、知识点

1. 客户端验证与服务器端验证

输入验证既可以发生在服务器端也可以发生在客户端，在检查用户输入的错误时，客户端验证可以在提交之前发现输入错误，避免对服务器请求与响应的往返过程，提高了 Web 窗体的可用性。

如果由开发人员来写客户端验证，常需要编写多个版本的代码来支持服务器和不同的浏览器，非常耗费时间。ASP.NET 的验证控件解决了这个问题，客户端验证与服务器端验证均支持 ASP.NET 的验证控件。因为验证逻辑被封装在控件中，所以这些控件创建了针对特定浏览器的代码，如果使用的浏览器支持客户端验证则可以进行客户输入验证，不支持脚本的浏览器不会接收客户端验证脚本。

在支持脚本的浏览器中，当用户单击“提交”按钮时，客户端验证就会发生，除非所有的客户端验证都通过，否则不会被回发到服务器。如果验证二次输入的密码不同于这类验证，则在一个输入控件转换到下一个输入控件时，就可以对已经完成的输入控件进行客户端验证。

ASP.NET 的验证控件都运行在服务器端，当 Web 页被发回到服务器时，服务器端重复执行客户端验证。这些重复验证可以防止用户绕过客户端脚本，使用一些输入进行欺骗，保证了数据的安全。服务器端输入验证除了一些格式验证，还可以用来比较用户输入的数据与已存储的数据是否匹配等多方面的验证。

2. 添加和设置验证控件

验证控件用于验证关联的输入控件中的用户输入。当用户输入的值未通过验证时，验证控件将显示错误消息。由于验证控件与输入控件是分开的，因此可以将错误消息定位在页面上相对于输入控件的任意位置。ASP.NET 提供了一些验证控件来执行特定类型的验证。

添加验证控件比较简单，打开 Web 窗体工具箱，选择一个需要的验证控件，然后把这个控件拖入要验证的输入控件旁边即可，然后打开“属性”窗口，进行属性设置。每个验证控件既有共有属性也有一些是特有的属性。下面介绍一些常用的共有属性：

✧ ControlToValidate 属性：用于设置或获取要关联的输入控件的 ID。可以通过本属性将多个验证控件关联到一个输入控件，只有输入控件所关联的验证控件的值都为真时，Web 窗体才能接受并处理输入控件。通过在本属性旁边的下拉列表中选择适当的输入控件 ID，来选择要进行验证的输入控件，该属性是验证控件必须要进行设置的属性，如果不设置会出现错误；

✧ EnableClientScript 属性：用于指示是否启用客户端验证，属性的默认值为 true。如果想要在验证发生前执行一些服务器代码时，可以通过设置本属性禁用客户端验证；

✧ ErrorMessage 属性：获取或设置要用于错误消息的文本。如果没有设置 Text 属性，则当验证控件被触发时，本属性是显示在验证控件位置上的错误信息。如果 Web 窗体使用了 ValidationSummary 控件，那么这个信息也会被包含在 ValidationSummary 控件中；

✧ Text 属性：用于指定验证失败时在验证控件中显示的文本。如果同时设置 ErrorMessage 属性和 Text 属性，当验证控件被触发时，Text 属性是显示在验证控件位置上的备份文本。如果使用 ValidationSummary 控件显示页面中所有验证失败的信息，ErrorMessage 属性在 ValidationSummary 控件中显示，而 Text 属性则仍旧显示在验证控件位置上；

✧ Display 属性：用于设置验证控件中错误消息的显示行为，本属性主要影响验证控件上错误信息的位置，但 ValidationSummary 控件上显示的信息不受本属性的影响。Display 的属性值有三个选项，即 Static 值为错误信息定义固定的布局，即使没有可见的错误信息文本验证控件也将占用空间；none 值表示在验证控件的位置上不显示错误信息；Dynamic 值使验证控件作为文本流的一部分呈现在页面上，当验证控件没有被触发时使用这个选项，而且不会在页面上显示空格，当显示错误信息时可能导致 Web 页面上 UI 元素发生位置变动。

3. 组合使用验证控件

有时候单一的验证控件无法充分验证用户的输入数据是否正确，因此可以使用多个验证控件与单个验证控件一起来验证不同的判断。

例如，在前面的练习中，用户名的输入不仅要满足格式要求，而且还要检查数据库中是否已经存在，同时还是必须输入的数据。这样，这个输入控件不仅需要与 RequiredFieldValidator 控件链接，同时还需要一个 RegularExpresionValidator 控件和一个 CustomValidator 验证控件。在对注册页面的输入验证中就有多个控件使用了组合验证。

4. 指定验证组

在一个页面中有可能需要分成不同情况的验证，这就需要使用验证组将页面上的某些验证控件归为一组。可以对每个验证组执行验证，该验证与同一页的其他验证组无关。将要分组的所有控件的 ValidationGroup 属性设置为同一个名称即可创建验证组，也可以为验证组分配任意名称，但该组的所有成员必须使用相同的名称。

在回发过程中，只根据当前验证组中的验证控件来设置 Page 类的 IsValid 属性，当前验证组是由导致验证发生的控件确定的。例如，如果单击验证组为 Group2 的按钮控件，并且其 ValidationGroup 属性设置为 LoginForm 的所有验证控件都有效，则 IsValid 属性将返回 true。对于其他控件（如 DropDownList 控件），如果控件的 CausesValidation 属性设置为 true，同时 AutoPostBack 属性设置为 true，则也可以触发验证。

注意：如果一个页面上有多个回发控件，而有些控件的回发行为发生时，不希望引发验证行为，则可以将这个控件的 CausesValidation 属性设置为 false。

5. ASP.NET 服务器控件的验证错误信息布局

当验证发生时，错误信息出现在页面上时，会成为页布局的一部分并可能更改页面的布局，而且影响页面的美观，因此需要在设计页的布局时适当放置可能出现的任何错误文本。

如果浏览器支持客户端验证，在显示客户端验证的错误信息文本时，可以通过设置验证控件的 Display 属性；如果浏览器不支持客户端验证，可以改为在一个表单元格或 Panel 控件中对验证控件进行布局。

在对错误信息布局时，可以通过如下四种不同的方式显示错误信息文本：

✧ 内联：在控件旁边验证控件所在的位置显示错误信息；

✧ 摘要：在一个涵盖所有错误的单独摘要中显示错误信息，该方式只在用户提交页时可用。这种方式可以在验证控件的 ErrorMessage 属性中包含详细的错误消息，并将

ValidationSummary 控件添加到页。详细的 ErrorMessage 属性文本将出现在页中 ValidationSummary 控件的位置上；

✧ 内联和摘要：同一错误信息的摘要显示和内联显示可能会有所不同，可以使用此选项内联显示较为简短的错误信息，而在摘要中显示较为详细的信息，也可以在输入字段旁显示错误标志符号，而在摘要中显示错误信息；

✧ 自定义：可以创建自己的错误信息显示。

6. Page.IsValid 属性

该属性验证控件测试用户的输入，设置错误状态并产生错误信息，但是验证控件不会改变程序的处理流程。如果发现用户的输入错误，验证控件不会绕过代码，所以在执行程序逻辑时，应该测试所有控件的状态，发现错误就立即终止代码运行，页面继续处理并将错误信息返回给用户。

为了在控件执行操作之前确定页面上所有的验证控件都有效，.NET 通过对 Page.IsValid 属性进行验证。

IsValid 属性对页面上的所有验证控件的值进行逻辑“与”运算，如果任何一个验证控件无效，则这个属性的返回值都为 false。这个属性提供了一个简单的方法来确定 Web 窗体上所有的输入控件是否有效，以及 Web 窗体是否已经准备继续处理业务逻辑。

如在注册页面中，可以在注册信息上传之前添加如下代码：

```
protected void btnCertain_Click(object sender, EventArgs e)
    {
        if (Page.IsValid)
       {
          //注册信息上传逻辑
       }
     }
    }
```

7. ASP.NET 验证控件

1）RequiredFieldValidator 必须字段验证控件

该控件将输入控件设置成为必选字段。通过在页面中添加 RequiredFieldValidator 控件并将其链接到必需的控件，可以指定某个用户在 ASP.NET 网页上的特定控件中必须提供的信息。如可以指定用户在提交注册窗体之前必须填写“姓名”文本框。该控件使用的标准代码如下：

```
<ASP:RequiredFieldValidator id="Validator_Name" Runat="Server" ControlToValidate=" 要检查的控件名 " ErrorMessage=" 出错信息 " Display="Static|Dymatic|None"></ASP: RequiredFieldValidator >
```

RequiredFieldValidator 控件有一个比较特殊的属性 InitialValue，属性值为一个字符串，默认值为 String.Empty，用来指定关联的输入控件的初始值，如果对这个属性设置值，则当输入控件中的值与该属性值相等时，验证失败。

2）CompareValidator 比较验证控件

该控件将对由用户输入到输入控件的值与输入到其他输入控件的值或常数值进行比较。该控件使用的标准代码如下：

```
<ASP:CompareValidator id="Validator_ID" RunAt="Server" ControlToValidate="要验证的控件 ID" errorMessage=" 错误信息 "ControlToCompare=" 要比较的控件 ID" Type="String|Integer|Double|DateTime|Currency" Operator="Equal|NotEqual|GreaterThan|GreaterTanEqual|LessThan|LessThanEqual|DataTypeCheck" Display= "Static|Dymatic|None"></ASP:CompareValidator>
```

其属性如下：

✧ Type 属性：表示要比较的控件的数据类型；

✧ Operator 属性：表示比较操作，有 7 种比较方式，即 Equal（相等）、NotEqual（不等）、GreaterThan（大于）、GreaterTanEqual（大于等于）、LessThan（小于）、LessThanEqual（小于等于）、DataTypeCheck（对输入控件的值与 Type 属性指定的数据类型之间的数据类型进行比较，如果无法将该值转换为指定的数据类型，则验证失败）；

✧ ControlToCompare 属性：设置指定要与之比较控件的 ID 号。

在使用时要注意 ControlToValidate 属性和 ControlToCompare 属性的区别，如果 operate 为 GreateThan，那么，ControlToCompare 属性值必须大于 ControlToValidate 属性值才是合法的。

比较验证控件既可以将输入控件的值与另一个输入控件的值进行比较，也可以将其与常数值进行比较。通常情况下不要同时设置 ControlToCompare 和 ValueToCompare 属性。如果同时设置了这两个属性，则 ControlToCompare 属性优先，ValueToCompare 属性会失去设置的意义。

3）RangeValidator 范围验证控件

验证输入是否在一定范围，该控件使用的标准代码如下：

```
    <ASP:RangeValidator id="Vaidator_ID" Runat="Server" controlToValidate="要验证的控件 ID"type="Integer" MinimumValue="最小值" MaximumValue="最大值" errorMessage="错误信息" Display="Static|Dymatic|None" >
    </ASP:RangeValidator>
```

其属性如下：

✧ MinimumValue 和 MaximumValue 属性：界定控件输入值的范围；

✧ Type 属性：定义控件输入值的类型。

水温输入值的范围控件代码如下：

```
    <asp:RangeValidator id="Range1" ControlToValidate="TextBox1" MinimumValue="0" MaximumValue= "100" Type="Integer" Text="水温值只能在 0～10!" runat="server"/>
```

4）RegularExpresionValidator 正则表达式验证控件

正则表达式验证控件的功能非常强大，当需要检查用户输入是否匹配一些规定的格式时，如电话号码、邮编、电子邮件地址等，就可以使用该控件进行验证。该控件使用的标准代码如下：

```
    <ASP:RegularExpressionValidator id="Validator_ID" RunAt="Server" ControlToValidate="要验证控件名" ValidationExpression="正则表达式" errorMessage="错误信息" display="Static|Dymatic|None" >
    </ASP:RegularExpressionValidator>
```

其中的 ValidationExpression 属性是重点。不同的字符表示不同的含义，常见正则表达式的字符及含义如表 2-9 所示。

表 2-9　常见正则表达式的字符及含义

字　符	含　义
.	表示任意字符
*	至少匹配前面表达式 0 次
+	至少匹配前面表达式 1 次
?	匹配前面表达式 0 次或 1 次

续表

字　符	含　义
|	匹配前面表达式或后面表达式
^	匹配字符串开头
$	匹配字符串结尾
[……]	匹配括号中的任何一个字符
[^……]	匹配不在括号中的任何一个字符
\w	匹配任意一个单词字符（a～z、A～Z、0～9 和下画线）
\W	匹配任意一个非单词字符
\s	匹配任意一个空白字符，包括制表符、换行符、回车符、换页符和垂直制表符
\S	匹配任意一个非空白字符
\d	匹配任意一个从 0～9 的字符
\D	匹配任意一个非数字的字符
[\b]	匹配一个退格键
{n,m}	匹配前面的字符最少 n 次，最多 m 次
{n,}	匹配前面的字符最少 n 次或更多次
{n}	匹配前面的字符 n 次
(……)	在单元中组合项目

下面列举几个常用的正则表达式：

✧ 验证是否为中文字符：[\u4e00-\u9fa5]；

✧ 验证空白行：\n\s*\r；

✧ 验证首尾空白字符：^\s*|\s*$；

✧ 验证网址 URL：[a-zA-z]+://[^\s]*；

✧ 验证国内电话号码：\d{3}-\d{8}|\d{4}-\d{7}；

✧ 验证腾讯 QQ 号：[1-9][0-9]{4,}；

✧ 验证中国邮政编码：[1-9]\d{5}(?!\d)；

✧ 验证身份证：(^\d{15})|(^\d{17}([0-9]|X)$)。

5）CustomValidator 自定义验证控件

以上给出的验证控件常常是无法满足需求的，这时就需要定义一个自定义的服务器端验证函数，然后使用 CustomValidator 控件来调用它。该控件使用的标准代码如下：

```
    <ASP:CustomValidator id="Validator_ID" Runat="Server" controlToValidate="要验证的控件" onServerValidate="验证函数"  errorMessage="错误信息" Display="Static|Dymatic|None" >
    </ASP: CustomValidator >
```

其属性如下：

✧ ClientValidationFunction 属性：获取或设置用于验证的自定义客户端脚本函数的名称；

✧ Onservervalidate 属性：设置在服务器端运行函数的名称。

CustomValidator 控件通常被用于类似用户名验证的情形，将用户输入的用户名和存储在数据库中的密码进行比较。它可以在服务器端进行验证，如果浏览器支持，也可以在客户端进行验证。与其他验证控件不同的是，该控件必须为 CustomValidator 控件编写验证代码。

检查输入控件中的数字是否为偶数。CustomValidator 控件代码：

```
    <asp:CustomValidator ID="CustomValidator1" runat="server" ErrorMessage="
CustomValidator" onservervalidate="CustomValidator1_ServerValidate" ClientVali
dationFunction="myclientfun"></asp:CustomValidator>
```

客户端代码：

```
    <script language="jscript" type="text/jscript">
        function myclientfun(source,arguments)
        {
          alert("这是客户端验证");
          var x=arguments.value;
          if(x%2==0) {arguments.IsValid=true;return;}
          else {arguments.IsValid=false;return;}

        }
       </script>
```

服务器端代码：

```
    protected void CustomValidator1_ServerValidate(object source, Server
ValidateEventArgs args)
       {
          int x = Convert.ToInt32(args.Value);
          if (x % 2 == 0) args.IsValid = true;
          else args.IsValid = false;
       }
```

通过编写 JavaScript 函数，重复服务器端方法的逻辑，从而添加客户端验证，在提交页面之前检查用户输入内容。即使使用了客户端检查，也应该执行服务器端的验证。服务器端的验证有助于防止用户通过禁用或更改客户端脚本来避开验证。

控件的 ServerValidate 事件可以创建一个基于服务器的事件处理程序，该事件将被调用来执行验证。source 参数是对引发此事件的自定义验证控件的引用。args.Value 属性包含要验证的用户输入内容。如果值是有效的，则将 args.IsValid 设置为 true；否则设置为 false。

6）ValidationSummary 验证信息显示控件

该控件主要用于收集本页的所有验证错误信息，并可以将它们组织以后再显示出来。该控件使用的标准代码如下：

```
    <ASP:ValidationSummary id="Validator_ID" RunAT="Server" HeaderText="头信息
"ShowSummary= "true|false" DiaplayMode="List|BulletList|SingleParagraph">
    </ASP: ValidationSummary >
```

其属性如下：

✧ HeaderText 属性：相当于表的 HeadText，用于设置或读取显示在摘要上方的文本；

✧ DisplayMode 属性：表示错误信息显示方式。List 每条错误信息都显示在单独的行中，相当于在每条信息后面添加
标记；BulletList 以项的方式显示，相当于 HTML 中的<LI>；SingleParegraph 每条错误信息都显示为段落中的一个句子；

✧ ShowMessageBox 属性：可以设置或读取错误信息是否在消息框中以弹出的方式进行显示。默认情况下该属性值为 false；

✧ ShowSummary 属性：设置或读取是否以内联方式显示验证信息，默认为 ture。如果 ShowMessageBox 属性与 ShowSummary 属性都设置为 ture，则在消息框和页面摘要中都显示验证信息。

ValidationSummary 验证控件本身不能验证数据，但可以用来显示其他验证控件的验证结果，在使用该控件之前需要先设置好其他验证控件的 ErrorMessage 属性。

ValidationSummary 验证控件常用方式代码如下，如图 2-21 所示是应用的结果显示。

```
    <asp:ValidationSummary ID="ValidationSummary1" runat="server" ShowMessage
Box="True" ShowSummary="true" HeaderText="错误信息汇总" ValidationGroup="Group1"
Width="160px" />
```

8. 自定义验证控件

为了提高 Web 开发的灵活性，ASP.NET 内置了一个可扩充的验证框架，该框架定义了服务器端和客户端的基本实现规则。使用这个可扩充的验证框架，可以根据自己的验证需要设计验证控件。

图 2-21 ValidationSummary 验证应用的结果显示

组成验证框架的主要有三个对象。

1）IValidator 接口

IValidator 接口是验证框架的基础，任何实现该接口的类都可以作为验证程序。IValidator 接口的定义代码如下：

```
public interface IValidator
{
    string ErrorMessage { get; set; }
bool IsValid { get; set; }
void Validate();
}
```

IValidator 接口包含以下两个属性和一个方法：

✧ ErrorMessage 属性：由类实现时，获取或设置条件验证失败时生成的错误消息；

✧ IsValid 属性：由类实现时，获取或设置一个值，通过该值指示用户在指定控件中输入的内容是否通过验证；

✧ Validate 方法：由类实现时，将计算它检查的条件，然后更新 IsValid 属性。

如果只是为了实现一个验证程序，而非验证控件，那么可以通过该接口完成；如果希望实现自定义验证控件，则需要继承 BaseValidator 类来实现。

2）BaseValidator 类

BaseValidator 类是验证框架中的核心部分，该类是一个抽象类，不能进行实例化。该类继承自 TextControl 基类，因此具有 Text 属性，并且实现 IValidator 接口。对于无效项，验证程序将显示 Text 属性的值。如果未设置该属性，验证程序将改为显示 ErrorMessage 属性的值，这就是在 ValidationSummary 控件中显示的文本。此行为与 Web 窗体验证程序一致。

无论是内置的验证控件，还是自定义的控件，都必须派生自 BaseValidator 类，可以实现所有验证控件都必须实现的通用属性，即 controltovalidate、ErrorMessage、Display、IsValid。由 BaseValidator 类派生的控件，可以不必再次实现以上通用属性，而只要根据应用需要另外定义一些属性和验证逻辑即可。

如代码清单 2-12 所示演示了一个验证输入长度的自定义验证控件，在项目中添加一个名为 customValidatecontrol.CS 的类。

代码清单 2-12　验证输入长度的自定义验证控件代码

```
using System.Web.UI.WebControls;
namespace myControls
{
    public class customValidatecontrol : BaseValidator
    {
        int _maxnumlength = 0;
        int _minnumlength = 0;
        public int Maxnumlength
        {
            get { return _maxnumlength; }
            set { _maxnumlength = value; }
        }
        public int Minnumlength
        {
            get { return _minnumlength; }
            set { _minnumlength = value; }
        }
        protected override bool EvaluateIsValid()
        {
          string value = this.GetControlValidationValue(this.ControlToValidate);
           if (value.Length > _maxnumlength || value.Length < _minnumlength)
           {  return false;  }
           else
           {   return true;     }
        }
    }
}
```

要在页面中使用自定义的验证控件，首先在页面中输入如下指令：

```
<%@ Register TagPrefix="cusvacon" Namespace="myControls" %>
```

然后在页面的合适位置添加自定义控件代码如下：

```
    <cusvacon:customValidatecontrol  ID="cuv1"  ValidationGroup="Group2"  Maxnum
length="20" Minnumlength="10" ErrorMessage="10～20 的字符串" ControlToValidate="
txtaddress" runat="server">
    </cusvacon:customValidatecontrol>
```

BaseValidator 类简化了自定义验证控件的过程，为控件开发人员提供了方便。

3）CustomValidator 类

CustomValidator 类派生自 BaseValidator 类，是五个内置控件之一。通过定义它的 ServerValidate 事件逻辑可以实现自定义服务器端验证，通过设置 ClientValidationFunction 属性可以完成客户端验证。CustomValidator 类不提供利用机制，因此不能被派生。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 5

拓展任务号	2-5	任务名称	为添加联系人页面添加输入验证
计划用时	30 分钟	任务性质	课外
任务描述与目标			
添加联系人页面中的输入数据也需要进行验证，通过对添加联系人页面添加输入验证加强对输入验证的理解与掌握			
主要操作步骤提示			

1．打开通讯录网站中的 addcontactor.aspx 文件；

2．为每个输入控件添加合适的验证控件，参考设计界面如图 2-22 所示。电话和移动电话规则验证的正则表达式分别是^[0]\d{3}-\d{8}|^[0]\d{3}-\d{7}|^[0]\d{2}-\d{8}和^[0]\d{11}|\d{11}。

图 2-22　加验证的添加联系人页面

2.2.4　ASP.NET 用户控件

在 ASP.NET 网页中除了可以使用 Web 服务器控件外，还可以使用用于创建 ASP.NET 网页的相同技术创建可重复使用的自定义控件。这类控件被称为用户控件，用户控件是一种复合控件，创建完成后可以被 Web 窗体作为服务器控件导入 ASP.NET 页。

一、实战演练

创建登录用户控件

（1）在“解决方案资源管理器”窗口中右击项目名，在弹出的快捷菜单中选择“添加新项”选项。在“添加新项”对话框中选择“Web 用户控件”，把名称改为“login.ascx”。

（2）切换到“设计”视图，创建用于保存控件的表，即布局表格。在表下拉菜单中选择“插入表”选项，使用“插入表”对话框创建带有三行和两列的表格，然后单击“确定”按钮。

（3）修改表的格局，选中表格中第三行的两个单元格，在表下拉菜单中执行“修改”→“合并单元格”命令。

（4）从工具箱的“标准”组中，将以下控件拖到表中，然后按如表 2-10 所示的内容设置它们的属性。

表 2-10　控件及其属性

控　件	属　性
拖动一个“label”控件到表的第 1 行第 1 列左列	ID：lstUserName TEXT：用户名
拖动一个“label”控件到表的第 2 行第 1 列左列	ID：lstPassWord TEXT：密码
拖动一个“TextBox”控件到表的第 1 行第 2 列左列	ID：txtUserName
拖动一个“TextBox”控件到表的第 2 行第 2 列左列	ID：txtPassWord
拖动两个“ImageButton”控件到表的第 3 行	设置 ImageUrl 属性值，用图片作为个性化按钮，作为“登录”和“注册”按钮

（5）对用户控件进行合适的样式设置后如图 2-23 所示。

（6）打开 login.ascx 的代码隐藏页，添加两个 ImageButton 图片路径属性，这两个属性用来读/写 ImageButton 控件的显示图片路径。

添加一个 StrPath 属性用来读/写设置用户单击“注册”按钮后的跳转路径。首先为 login 类添加一个字符串变量，用来存放跳转路径，并给这个变量设置默认值。login.ascx.cs 代码如代码清单 2-13 所示。

图 2-23　登录用户控件

代码清单 2-13　login.ascx.cs 代码。

```
public partial class login : System.Web.UI.UserControl
{
    string strpath="regist.aspx";
    public string EnterImgpath
    {
        set { this.ImageButton1.ImageUrl = value; }
        get { return this.ImageButton1.ImageUrl; }
    }
    public string RegistImgpath
    {
        set { this.ImageButton2.ImageUrl = value; }
        get { return this.ImageButton2.ImageUrl; }
    }
    public string StrPath
    {
        set { strpath = value; }
        get { return strpath; }
    }
}
```

（7）新建一个 Web 页面“Default.aspx”，在“解决方案资源管理器”中将建好的用户控件拖到 Web 页面中，并通过“属性”面板修改用户控件的属性。并对“Default.aspx”作适当的美化后在浏览器查看运行结果。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-11 所示。

表 2-11 演练完成情况评价表

任 务 号	2-4	任务名称	创建登录用户控件
任务子项	完成情况	主要问题	未完成原因
创建用户控件			
创建用户控件属性与事件			
使用用户控件			

三、知识点

1. 用户控件概述

用户控件与 ASP.NET 页十分相似，可以在用户控件上使用与在 ASP.NET 网页上相同的 HTML 元素（html、body 和 form 元素除外）和 Web 控件，它包含若干控件及处理事件的代码。用户控件与 ASP.NET 网页有以下区别：

✧ 用户控件的文件扩展名为.ascx；

✧ 用户控件中没有@ Page 指令，而是包含@ Control 指令，该指令对配置及其他属性进行定义，如<%@ Control Language="C#" AutoEventWireup="true" CodeFile="login.ascx.cs" Inherits="login" %>；

✧ 用户控件不能作为独立文件运行，而必须像处理控件一样，将它们添加到 ASP.NET 页中；

✧ 由于用户控件是被 Web 窗体页面引用的，因此用户控件中没有 html、body 和 form 元素，这些元素必须位于宿主页中。

用户控件一旦被创建，就能在整个项目的 Web 窗体中使用。然而因为没有将用户控件编译成程序集，所以在每个使用控件的 Web 应用程序项目中，都必须保存一个该控件的副本；也无法在工具箱中加载用户控件，当使用用户控件时要通过“解决方案资源管理”将其拖放到 Web 窗体页来创建。

使用用户控件的 Web 页面称为该控件的宿主页，用户控件必须被放入宿主页中才可以进行调试运行。

2. 把现在的 Web 窗体页转换为用户控件

用户控件可以通过新建方式进行创建，也将 Web 页面更改为一个用户控件。

首先把 Web 的扩展改为.ascx，如果是使用代码隐藏页，则根据使用的编程语言，重命名代码隐藏文件使其文件扩展名为.ascx.cs，然后打开代码隐藏文件并将该文件继承的类从 Page 更改为 UserControl。打开.aspx 文件，在“源”视图中移除 html、body 和 form 元素，将@ Page 指令更改为@ Control 指令，移除@ Control 指令中除 Language、AutoEventWireup（如果存在）、CodeFile 和 Inherits 之外的所有属性，在@ Control 指令中将 CodeFile 属性更改为指向重命名的代码隐藏文件。

3. 定义用户控件的属性和事件方法

宿主页不能直接访问用户控件的 UI 元素，而是通过设置 GET 和 SET 属性来定义用户控件的公共属性，才能在宿主页面中使用或设置用户控件的 UI 元素值。

用户控件的事件方法创建与页面的事件方法创建的方式相同，创建后就可视为是该控件在宿主页中的方法。

4. 通过@ Register 指令在 Web 页中添加用户控件

通过此方式添加用户控件一般来说要经过三个步骤：

（1）在 ASPX 页面中创建 @ Register 指令，然后分别设置如下属性：

✧ TagPrefix 属性：将前缀与用户控件相关联。此前缀将包括在用户控件元素的开始标记中；

✧ TagName 属性：将名称与用户控件相关联。此名称将包括在用户控件元素的开始标记中；

✧ Src 属性：定义包括用户控件文件的虚拟路径。注意，Src 属性值既可以是相对路径，也可以是从应用程序的根目录到用户控件源文件的绝对路径。为灵活使用，建议使用相对路径。“~”表示应用程序的根目录。用户控件不能位于 App_Code 目录中，如：

```
<%@ Register Src="login.ascx" TagName="login" TagPrefix="uc1" %>
```

（2）在网页主体 form 元素内部声明用户控件元素，如：

```
<uc1:login ID="login1" runat="server" />
```

（3）设置用户控件公开公共属性。可以在“属性”面板中设置，也可以通过编程方式进行设置，设置方式与 ASP.NET 的内置控件设置方式相同。

可以通过在“解决方案资源管理器”中拖动.ascx 文件到页面指定位置的方法添加用户控件，这种方法其实是简化了前面两个步骤。

5. 通过编程的方式在 Web 页面中添加用户控件

和 ASP.NET 网页上创建任何服务器控件的一样，也能以同样的方式创建用户控件的实例，但以编程方式添加用户控件比较复杂。

（1）在用户控件的 @ Control 指令中添加 ClassName 属性，以确保将类名分配给用户控件，如：

```
<%@ Control Language="C#" ClassName="logincon" AutoEventWireup="true"
CodeFile= "login.ascx.cs" Inherits="login" %>
```

（2）在要添加用户控件的 Web 页面的 ASPX 文件中，通过 @ Reference 指令创建对该用户控件的引用。当通过编程方式创建用户控件时，只有创建了对该控件的引用之后，ASP.NET 网页才能使用该用户控件的强类型，如：

```
<%@ Reference Control="login.ascx" %>
```

（3）在 Web 页面的逻辑代码中，使用用户控件的类名创建该控件的实例变量，该类将成为 ASP 命名空间的一部分，如：

```
protected ASP.logincon login1;
```

（4）通过在代码中调用 LoadControl 方法创建用户控件的实例，然后根据需要分配属性值，并向页容器中添加控件，如：

```
protected void Page_Load(object sender, EventArgs e)
    {
        login1 = (ASP.logincon)LoadControl("~/login.ascx");
        login1. StrPath ="regist.ascx";
        Panel1.Controls.Add(login1);
}
```

6. 使用用户控件的优点

在 ASP.NET 的 Web 应用程序中使用用户控件有多种用途，如创建导航条、页脚等，也可以用于重复使用的代码块。其优点如下：

✧ 用户控件是自包含的，它们提供了单独的变量命名空间，这意味着用户控件的方法和属性不会与宿主页面中任何已存在的方法和属性冲突;

✧ 用户控件可以在宿主页内多次使用而不会引起属性和方法的冲突;

✧ 用户控件和宿主页面的编写语言可以不同。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 6

拓展任务号	2-6	任务名称	导航条用户控件
计划用时	30 分钟	任务性质	课外

任务描述与目标

导航条是一个网页的组成部分，一个网站不同页面的导航条无论是布局、色彩和功能基本相同，将其创建成用户控件可以增强代码重用。本任务实现为网络通讯录的后台管理创建导航条的用户控件，结合对知识点的学习，增强对用户控件系统性的认识

主要操作步骤提示

1．在资源管理器中添加一个用户控件文件；

2．在“设计”视图中添加按钮图标，给这两个按钮图片设置超级链接，并设置相关属性；

3．注意在不同目录中产生的链接地址会不同，因此，为导航的地址添加属性；

4．参考代码如下：

```
string a1herf, a2href, a1target, a2target;
    public string A1Url
    {
        set { a1.HRef = value; }
        Get { return a1.HRef; }
    }
    public string A2Url
    {
        set { a2.HRef = value; }
        get {return a2.HRef; }
    }
    public string A1Target
    {
        set { a1.Target = value; }
        get {return a1.Target; }
    }
    public string A2Target
    {
        set { a2.Target = value; }
        get {return a2.Target; }
    }
```

2.3 任务 2 ASP.NET 内部对象

任务描述

在 Web 应用程序运行时，ASP.NET 将维护有关当前应用程序、每个用户会话、当前 HTTP 请求、请求的页等方面的信息。ASP.NET 包含一系列类，用于封装这些上下文信息，所在命名空间：System.Web。ASP.NET 中这些类的实例可用作代码访问的内置对象。ASP.NET 内置对象包括：Response、Request、Context、Server、Application、Session、Trace。其中 Response、

Request、Server、Session 这个四个内置对象可通过 HttpContext.Current 或 Page 来获取，本节将学习这四个内置对象的使用方式。

本任务通过对网络通讯录各页面间的链接来完成ASP.NET内部几个重要内置对象的学习。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）了解什么是 ASP.NET 内置对象；

（2）掌握 ASP.NET 内置对象的使用方法和工作原理。

2.3.1 页面间跳转

经过上面几节的学习，完成了网站通讯录的页面设计，其中添加联系人页面、查找页面和详细信息页面要经过用户登录后才能使用，管理页面负责这些页面的导航。

一、实战演练

（1）打开“cpmanage”在文件夹中的“cpmanage.aspx”Web 文件，在切换到“源视图”，修改“源视图”代码。修改后“cpmanage.aspx”的部分代码如下：

```
    <table id="menu">
    <tr>
      <td><a  href="addcontactor.aspx"  target="if1"><img  alt=""  src="iamges1/
bt1.jpg" /></a> </td>
      <td>  <a  href="fingcp.aspx"  target="if1">  <img  alt=""  src="iamges1/
bt3.jpg" /></a></td>
    </tr>
    </table>
```

（2）打开用户控件 login.ascx，选中“注册”按钮双击，或在“事件”面板中双击“Click”事件。在事件中添加代码，代码如下：

```
    protected void ImageButton2_Click(object sender, ImageClickEventArgs e)
        {
            Response.Redirect("cpmanage /cpmanage.aspx");
        }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-12 所示。

表 2-12　演练完成情况评价表

任　务　号	2-8	任务名称	页面间跳转
任务子项	完成情况	主要问题	未完成原因
创建联系人管理页面			
添加允许访问事件代码			

三、知识点

1. Web 页面的重定向

Web 应用程序经常需要实现页面间跳转，即页面的重定向，ASP.NET 提供了多种重定向

的方式，重定向引发与服务器的交互。

HTTP 定义了与服务器不同的交互方式，其中 GET 和 POST 是两种最基本的交互方式。一般在浏览器中输入地址都是通过 GET 方式，在 ASP.NET 的服务器控件提交为 POST 方式。ASP.NET 调用网页的方式基本上包括：原始请求（GET）、回发（POST）、跨页发送（POST）、服务器传输。这几种方式对 Page 类中 IsPostBack、PreviousPage、IsCrossPagePostBack 和 IsCallback 的属性值有不同的影响，也可以通过这几个属性值来判断页间的重定向方式。

1）原始请求

原始请求的方式很多，包括在浏览器的地址栏中输入原始地址、使用 HTML 创建的静态链接、使用 HyperLink 控件、通过 Response.Redirect()方法在服务器代码中控制超链接的链接文本和目标 URL。在此情况下目标页使用 GET 方式进行调用，因此，不会将与源页有关的任何信息传递到目标页，除非在目标页的 URL 上指定查询字符串。如果源页和目标页位于同一个 Web 应用程序中，可以使用会话状态或应用程序状态来共享信息。

通过这种方式调用的 Page 类的 IsPostBack 属性值为 false，PreviousPage 属性值为 null，IsCallback 属性值为 false。

2）回发

ASP.NET 在默认情况下，单击一些服务器控件引发页面中的信息会发送到 Web 服务器，然后该页面再次运行的过程为回发。回发方式下的 Page 类 IsPostBack 属性值为 true，PreviousPage 属性值为 null，IsCallback 属性值为 false。

3）跨页发送

在 ASP.NET 网页中引起回发事件一般把页直接提交回该页本身。在某些情况下，可能需要将一个页发送到其他页，通过设置按钮的 PostBackUrl 值可以实现跨页发送。跨页发送与超链接的类似之处在于通过用户操作来启动传输。但是，在跨页发送中，目标页是使用 POST 命令调用的，该命令会将源页上控件的值发送到目标页。此外，如果源页和目标页位于同一个 Web 应用程序中，则目标页可以访问源页的公共属性。

通过这种方式调用的 Page 类 IsPostBack 属性值为 false，PreviousPage 属性值为引用源页，Page.PreviousPage.IsCrossPagePostBack 为 true，IsCallback 属性值为 false。

跨页发送可以使用 FindControl 方法来搜索源页上的控件并提取这些控件的值，也可以获取源页的公共成员的值，最常见的方法是在源页定义公共属性，在目标页上获取这些属性的值。

4）服务器传输

通过调用 Transfer 方法，在服务器上以编程方式重定向到目标页。此方式中服务器只是将当前源页的上下文传输给目标页，然后目标页呈现在源页的位置。源页和目标页必须位于同一个 Web 应用程序中。与跨页发送一样，Transfer 方法也具有能够使目标页从源页中读取控件值和公共属性值的优点。

由于源页和目标页之间的传输在服务器上进行，浏览器没有任何关于更改后页的信息，它仍保留有关源页的 URL 信息。例如，浏览器中的地址栏在执行传输后不会发生变化，而是继续显示源页的 URL，也不会更新浏览器的历史记录以反映传输过程。如果用户在浏览器中刷新页面或单击浏览器的“后退”按钮，可能导致意外行为。因此，对于以隐藏 URL 的方式向用户呈现页面的应用程序而言，调用 Transfer 方法是一种比较好的方法。

通过这种方式调用的 page 类属性值与跨页发送方式不同的是 PreviousPage.IsCross

PagePostBack 为 false。

2. Response 对象

Response 对象可以动态地响应客户端的请求，并将动态生成响应结果返回给客户端浏览器。Response 对象是 HttpResponse 类的一个实例，可以用于向客户端输出数据、实现页面的跳转等。在非继承 Page 的 CS 文件完全限定名称为 HttpContext.Current. Response。

其常用的属性和方法如下：

✧ Buffer：获取或设置一个值，该值指示是否缓冲输出并在处理完整个响应之后发送；

✧ Cache：获取网页的缓存策略；

✧ Charset：获取或设置输出流的 HTTP 字符集；返回信息为 String 类型；

✧ ContentEncoding：获取或设置输出流的 HTTP 字符集；返回信息为 Encoding 类型实例，其中包含与当前响应的字符集有关的信息。

✧ HeaderEncoding：获取或设置一个 Encoding 对象，该对象表示当前标头输出流的编码；

✧ ContentType：获取或设置输出流的 HTTP MIME 类型；

✧ Cookies：获取响应 Cookie 集合；

✧ Expires：获取或设置在浏览器上缓存的页在过期之前的分钟数。如果用户在页面过期之前返回同一页，则显示缓存的版本；

✧ OutputStream：启用到输出 HTTP 内容主体的二进制输出。

✧ BinaryWrite()：将一个二进制字符串写入 HTTP 输出流；

✧ Clear()：清除缓冲区流中的所有内容输出；

✧ End()：将当前所有缓冲的输出发送到客户端，停止该页的执行；

✧ Flush()：向客户端发送当前所有缓冲的输出；

✧ Redirect()：将客户端重定向到新的 URL；

✧ SetCookie()：更新 Cookie 集合中的一个现有 Cookie；

✧ Write()：将信息写入 HTTP 响应输出流；

✧ WriteFile()：将指定的文件直接写入 HTTP 响应输出流；

3. Response 对象应用实例

Response 对象的方法和属性在实际应用比较广泛，下面代码清单 2-14 展示了 Response 常用属性的用法，从 FileUpload1 上传一个图片文件并把图片显示在网页上。

代码清单 2-14 把上传的图片显示在页面上

```
    <asp:FileUpload ID="FileUpload1" runat="server" />
    <asp:Button  ID="Button1"  runat="server"  OnClick="Button1_Click"  Text="
Button" />
    protected void Button1_Click(object sender, EventArgs e)
        {
            Response.ContentType = "image/jpeg";
            Response.Clear();
            Response.BufferOutput = true;
    //把上传图片的内容放入 Bitmap 对象中
            Bitmap bmp = new Bitmap(FileUpload1.FileContent);
            bmp.Save(Response.OutputStream, ImageFormat.Jpeg);
            bmp.Dispose();
            //图片信息显示在网页上
            Response.Flush();
    Response. End()
        }
```

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 7

拓展任务号	2-7	任务名称	为注册页面添加跳转功能
计划用时	10 分钟	任务性质	课外
任务描述与目标			
用户成功注册后，一般有两种处理。一种是回到登录页面让用户重新登录以增强记忆，另一种就是直接转入个人管理中心。			
主要操作步骤提示			
1．打开注册页面“regist.aspx”，双击"btnCertain"按钮，在“btnCertain_Click”事件中添加如下代码： `Response.Redirect("cpmanage /cpmanage.aspx");` 2．浏览查看事件结果。			

2.3.2 页间传值

当 Web 服务器接收到对某页的请求时，会找到该页，对其进行处理，将其发送到浏览器，然后丢弃所有页信息。如果用户再次请求同一页，服务器则会重复整个过程：从头开始对该页进行重新处理。换言之，服务器不会记忆它已处理的页，页是无状态的。应用程序需要维护有关某页的信息，其无状态的性质就成为了一个问题。Web 页间的信息传递则通过特殊的方法完成。

一、实战演练

（1）打开用户控件 login.ascx，选中“登录”按钮并双击，或在“事件”面板中双击“Click”事件。在事件中添加代码，代码如下：

```
    protected void ImageButton2_Click(object sender, ImageClickEventArgs e)
        {
            Response.Redirect("cpmanage   /cpmanage.aspx?uname="+   txtUserName.
Text);
        }
```

（2）打开“cpmanage.aspx”文件，在“设计视图”中双击空白处。在“Load”事件中添加如下代码：

```
    if(!Page.IsPostBack)
    {
        if (Request.QueryString["uname"] != null)
        {
          topic.InnerText  =  Request.QueryString["uname"].ToString()  +  topic.
InnerText;
        }
    }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-13 所示。

表 2-13　演练完成情况评价表

任　务　号	2-6	任 务 名 称	页面间跳转
任务子项	完成情况	主要问题	未完成原因
给用户控件 login.ascx 的“登录”按钮添加传值重定向			
在“cpmanage.aspx”文件中显示登录人用户名			

三、知识点

从应用程序的一个 Web 网页重定向到另一个页，经常希望将信息从源页传递到目标页，可以采用各种方式在页之间传递信息，某些方式取决于重定向的方式。

1. 页间信息传递常用的方法。

1）使用查询字符串

该字符串可将信息追加到目标页的 URL 上。使用 HyperLink 控件将导航内置于页中时，或使用 Redirect 方法以编程方式重定向到其他页时，都可以使用查询字符串。即使这些页不在同一个 Web 应用程序中，也能使用查询字符串传递值。如果要将信息传递到非 ASP.NET 网页的页中，也可以采取这种方式。如果目标页是一个 ASP.NET 网页，则可从 HttpRequest 对象的 QueryString 属性中读取查询字符串的值。不要使用查询字符串传递敏感数据，因为查询字符串中的信息对用户是可见的，而且很容易被修改，因此会带来潜在的安全风险。

2）使用会话状态可以存储信息

可从当前应用程序的所有 ASP.NET 网页中访问这些信息。但是，这种方法会占用服务器内存，并且其信息将在会话过期之前一直存储在内存中，因而与直接向下一页传递信息相比，其系统开销更大。

3）直接读取源页中的信息

当源页跨页发送到目标页或调用 Transfer 方法在服务器上执行从源页转到目标页时，可以采用以下方式实现页间信息传递，这种方式较少使用，代码如下：

```
TextBox UserName = (TextBox)LoginControl.FindControl("UserName");
```

2. Session 会话对象

在 ASP.NET 中，HttpSessionState 类提供对会话状态值及会话级别设置和生存期管理方法的访问。Session 会话对象是 HttpServerUtility 类的对公开象。

Session 会话对象主要用于存储单个用户的信息，不同的用户会产生一个属于自己的 SessionID 号。ASP.NET 会话状态能使当前用户在构成 Web 应用程序的不同 ASP.NET 页面之间导航，为用户存储检索值。ASP.NET 会话状态在有限时间段内从同一个浏览器接收到的请求标识为一个会话，并在该会话持续期间保留变量的值。默认情况下，所有的 ASP.NET 应用程序都启用 ASP.NET 会话状态。使用 Session 属性可方便地设置和检索 ASP.NET 会话状态变量。

在非继承 Page 的 CS 文件完全限定名称为 HttpContext.Current. Session。

HttpSessionState 类的常用属性和方法如下：

✧ CookieMode 属性：获取一个值，该值指示是否为无 Cookie 会话配置的应用程序；

✧ Item [([(Int32)])]属性：按数字索引获取或设置会话值；

✧ Item [([(String)])]属性：按名称获取或设置会话值；

✧ Keys 属性：获取存储在会话状态集合中所有键和值的集合；

✧ SessionID 属性：获取会话的唯一标识符；

✧ Timeout 属性：获取并设置在会话状态提供程序终止会话之前，各请求之间所允许的时间（以分钟为单位）；

✧ Abandon()方法：取消当前会话；

✧ Add()方法：向会话状态集合添加一个新项；

✧ Clear()方法：从会话状态集合中移除所有的键和值；

✧ Remove()方法：删除会话状态集合中的项；

✧ RemoveAll()方法：从会话状态集合中移除所有的键和值；

✧ RemoveAt()方法：删除会话状态集合中指定索引处的项。

3. Request 对象

读取客户端在 Web 请求期间发送的 HTTP 数据包的全部信息，提供以当前页请求的访问，包括标题、客户端证书、查询字符串等，是 HttpRequest 类的一个实例，在非继承 Page 的 CS 文件完全限定名称为 HttpContext.Current. Reques。

其常用的属性和方法如下：

✧ ApplicationPath：获取服务器上 ASP.NET 应用程序的虚拟应用程序根路径；

✧ Browser：获取或设置有关正在请求的客户端浏览器功能的信息；

✧ ContentLength：指定客户端发送的内容长度（以字节计）；

✧ ContentType：获取或设置传入请求的 MIME 内容类型；

✧ Cookies：获取客户端发送 Cookie 的集合；

✧ FilePath：获取当前请求的虚拟路径；

✧ Filter：获取或设置在读取当前输入流时要使用的筛选器；

✧ Form：获取窗体变量集合；

✧ QueryString：获取 HTTP 查询字符串变量集合；

✧ ServerVariables：获取 Web 服务器变量的集合；

✧ TotalBytes：获取当前输入流中的字节数；

✧ Url：获取有关当前请求的 URL 信息；

✧ UrlReferrer：获取有关客户端上次请求的 URL 信息，该请求链接到当前的 URL；

✧ UserHostAddress：获取远程客户端的 IP 主机地址；

✧ UserHostName：获取远程客户端的 DNS 名称；

✧ BinaryRead()：对当前输入流进行指定字节数的二进制读取；

✧ MapPath()：为当前请求将请求的 URL 中的虚拟路径映射到服务器上的物理路径；

✧ SaveAs()：将 HTTP 请求保存到磁盘。

Request 对象主要用于客户端发送的信息，其中也包括了一些客户端主机和浏览器的信息，下面代码段展示了 Request 对象获取客户端的一些主要信息。

```
string str = "";
HttpBrowserCapabilities bc = Request.Browser;
str = str + bc.Browser + "<br/>";//获取浏览器名
```

```
    str = str + bc.Version + "<br/>";//获取浏览器完整版本号
    str = str + bc.MajorVersion;//获取浏览器主版本号
    str = str + Request.UserHostAddress+ "<br/>";
    str = str + Request.UserHostName + "<br/>";
    Label1.Text = str;
```

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 8

拓展任务号	2-8	任务名称	为管理联系人所涉及的页面添加权限使用代码
计划用时	10 分钟	任务性质	课外

任务描述与目标

与联系人有关的页面只有在用户登录后才能访问，在相关页面导入浏览器前先查看用户是否已经登录。

主要操作步骤提示

1．打开用户控件 login.ascx，双击“登录”按钮，在“Click”事件中添加如下代码。

```
if (txtname.Text.Trim()="张三"&&txtpws.Text.Trim()="123456")
{
 Session["username"] = txtname.Text.Trim();
Page.Header.Title = txtname.Text + ",欢迎您回来！";
Response.Redirect(@"cpmanage\cpmanage.aspx");
}
else
Page.ClientScript.RegisterStartupScript(this.GetType(), null, "alert('用户名不存在或密码错，请重新登录！');", true);
```

2．打开“cpmanage”文件夹下所有与管理联系人信息相关的页面，在“Load”事件中添加如下代码：

```
if (Session["username"] == null)
   {
      Response.Redirect("../Default.aspx");
  }
```

3．运行“cpmanage”文件夹下的文件，发现只有在“Default.aspx”页面正确登录后才能访问。

2.3.3 服务器消息的获取

一、实战演练

（1）在“解决方案资源管理器”窗口中右击项目名，添加“pictures”文件夹。

（2）打开“regist.aspx”文件，双击“确定”按钮，或在“事件”面板中双击“Click”事件。在事件中添加代码，代码如下：

```
    protected void btnCertain_Click(object sender, ImageClickEventArgs e)
    {
        string filepath = Server.MapPath(@"~\");//"."也可以
        if (filupImage.HasFile)
        {
     filename = @"pictures\" + DateTime.Now.Year.ToString() +
              DateTime.Now.Month.ToString() + DateTime.Now.Day.ToString() +
              DateTime.Now.Hour.ToString() + DateTime.Now.Minute.ToString() +
```

```
                DateTime.Now.Second.ToString() +
filupImage.FileName.Substring(filupImage.FileName.LastIndexOf('.'));
                fileFull = filepath + @"\" + filename;
                filupImage.SaveAs(fileFull);
            }
        }
```

（2）运行文件，上传一个文件，查看“pictures”文件夹中的变化。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-14 所示。

表 2-14 演练完成情况评价表

任 务 号	2-7	任 务 名 称	页面间跳转
任务子项	完成情况	主要问题	未完成原因
给用户控件 login.ascx 的“登录”按钮添加传值重定向			
在“cpmanage.aspx”文件中显示登录人用户名			

三、知识点

ASP.NET 通过 HttpServerUtility 类提供获取服务器上信息的方法和属性，以及进行 HTML 编码的功能。HttpServerUtility 类的方法和属性通过 ASP.NET 提供的内部 Server 对象公开。在页面中直接通过 Server 对象访问，在非 Web 文件中如果需要以编程方式使用 Server 对象，必须指定完全限定名称 HttpContext.Current.Server。

1. Server 对象常用方法和属性

其相关属性和方法如下：

✧ MachineName：获取服务器的计算机名称；

✧ ScriptTimeout：获取和设置请求超时值（以秒计）；

✧Execute()：在当前请求的上下文中执行指定资源的处理程序，然后将执行返回给调用它的页；

✧Transfer():对于当前请求，终止当前页的执行，并使用指定的页 URL 路径来开始执行一个新页。

✧ HtmlDecode()：对已被编码、已消除无效 HTML 字符的字符串进行解码；

✧ HtmlEncode()：对要在浏览器中显示的字符串进行编码；

✧ MapPath()：返回与 Web 服务器上的指定虚拟路径相对应的物理文件路径；

✧ Transfer()：终止当前页的执行，并为当前请求开始执行新页；

✧ TransferRequest()：异步执行指定的 URL；

✧ UrlDecode()：对字符串进行解码，该字符串针对 HTTP 传输进行了编码并在 URL 中发送到服务器；

✧ UrlEncode()：编码字符串，以便通过 URL 从 Web 服务器到客户端进行可靠的 HTTP 传输；

✧ UrlPathEncode()：对 URL 字符串的路径部分进行 URL 编码并返回编码后的字符串。

2. 字符串的编码与解码

有时候，在传递参数时，是将数据附在网址后面传递，但是如果遇到一些如“#”等特殊字符的时候，就会读不到这些字符后面的参数。所以需要在传递特殊字符的时候，需将要传递的内容先以 UrlEncode 编码，这样才可以保证所传递的值可以被顺利读到。

另外有些服务器对中文不能很好的支持，这时候也需要利用 UrlEncode 对其进行编码，以被服务器所识别，代码示例如下：

```
    //HtmlEncode 对字符串进行 HTML 编码并返回编码后的字符串，所以这个原样输出
    Response.Write(Server.HtmlEncode("粗体标记为：<B>粗体文字</B>"));
    //HtmlDecode 对字符串进行 HTML 解码并返回解码后的字符串，所以这个后面加粗
    Response.Write(Server.HtmlDecode("粗体标记为：<B>粗体文字</B>"));
    //在 Default4.aspx 页面输出：name@
    Response.Write("<A href='Default4.aspx?data=name@#163.com'>没有编码的参数内
容</A><br>");
    //在 Default4.aspx 页面输出：name@#163.com
    Response.Write("<A href='Default4.aspx?data=" + Server.UrlEncode("name@#163.
com") + "'>编码的参数内容</A><br>");
```

3. Execute()与 Transfer()区别

Execute()与 Transfer()这两个方法在使用时有相同之处和不同之处。

Execute()方法可以在当前页面中执行同一 Web 服务器上的另一页面，当该页面执行完毕后，控制流程将重新返回到原页面中发出 Server.Execute 方法调用的位置，可以将一个.aspx 页面的输出结果插入到另一个.aspx 页面中。

Transfer()方法是终止当前页的执行，并将执行流程转入同一 Web 服务器的另一个页面。被调用的页面应是一个.aspx 页面，在页面跳转过程中，Request 等对象保存的信息不变，这意味着从页面 A 跳转到页面 B 后可以继续使用页面 A 中提交的数据。

此外，由于 Server.Transfer()方法调用是在服务器端进行的，客户端浏览器并不知道服务器端已经执行了一次页面跳转，所以实现页面跳转后浏览器地址栏仍将保存页面 A 的 URL 信息。因此无论是 Execute()还是 Transfer()在地址栏中都看不到地址的变化。

ASP.NET 不验证当前用户是否有权查看由这两个方法提交的资源。虽然 ASP.NET 授权和身份验证逻辑运行于调用原始资源处理程序之前，但 ASP.NET 仍将直接调用这两个方法指示的处理程序，并且不为新资源重新运行授权和身份验证逻辑。如果应用程序的安全策略要求客户端具有适当的授权才能访问相应的资源，则应用程序应强制再次授权或提供一种自定义访问控制机制。应用实例代码如下。

```
public partial class Default : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Response.Write("<P>调用 Execute 方法之前</P>");
        Server.Execute("Default2.aspx");//
        Response.Write("<P>调用 Execute 方法之后</P>");
        Response.Write("<P>调用 Transfer 方法之前</P>");
        Server.Transfer("Default2.aspx");//
        Response.Write("<P>调用 Transfer 方法之后</P>");
    }
}
```

2.4 任务 3 ADO.NET 连接环境下的数据库操作

任务描述

大部分应用程序都围绕读取和更新数据库中的信息进行操作，ASP.NET 通过 ADO.NET 提供的数据访问服务类完成数据的处理任务。

在本任务中通过对通讯录各功能的实现，学习如何设计一个网站的逻辑，以实现网站的功能。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）掌握 ADO.NET 的原理；

（2）能实现数据库的连接；

（3）实现对数据库的数据操作；

（4）能在网页中显示数据库的内容；

（5）根据网站的功能设计合理的逻辑代码。

2.4.1 连接数据库环境

一、实战演练

（1）在“解决方案资源管理器”窗口中双击“web.config”文件，在“<connectionStrings>”节中，添加如下代码：

```
    <connectionStrings>
    <add name="addressconn" connectionString="server=数据库服务器名; database=
addressbook; user id=用户名;Password=密码" />
    </connectionStrings>
```

（2）打开“regist.aspx.cs”注册页面的代码文件，在 CS 文件头添加命名空间，代码如下：

```
    using System.Data.SqlClient;
```

（3）在 CS 类内定义变量，代码如下：

```
    SqlConnection sqlconn;
```

（4）双击“确定”按钮，事件代码中添加数据库连接代码，如代码清单 2-15 所示。

代码清单 2-15 数据库连接代码

```
    protected void btnCertain_Click (object sender, EventArgs e)
    {
        try
        {
            sqlconn = new SqlConnection
           (ConfigurationManager.ConnectionStrings["addressconn"] . ToString());
            sqlconn.Open();
          Page.ClientScript.RegisterStartupScript(this.GetType(), null, "alert('
连接成功! ');", true);
        }
        catch
        { Page.ClientScript.RegisterStartupScript(this.GetType(), null, "alert('
```

```
连接不成功! ');", true);}
    if inally
          { if (sqlConn.State == ConnectionState.Open)  sqlConn.Close(); }
    }
```

（5）启动调试，如果弹出“连接成功”对话框则说明连接成功。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-15 所示。

表 2-15　演练完成情况评价表

任　务　号	2-8	任 务 名 称	创建数据库连接
任务子项	完成情况	主要问题	未完成原因
添加 web.config 数据配置			
创建连接			

三、知识点

如果建立的是根据不同用户请求显示不同信息的 Web 动态站点，就必须把 Web 应用程序同各种数据库源连接起来。ASP.NET 是通过 ADO.NET 组件所提供的数据访问类以编程的方式对数据源进行访问的。创建数据库连接是数据操作的起始点。

1. ADO.NET 概述

ADO.NET 是一组向.NET 程序员公开数据访问服务的类。 ADO.NET 为创建分布式数据共享应用程序提供了一组丰富的组件。 它提供了对关系数据、XML 和应用程序数据的访问。ADO.NET 支持多种开发需求，包括创建由应用程序、工具、语言或 Internet 浏览器使用的前端数据库客户端和中间层业务对象。

ADO.NET 包含用于连接到数据库、执行命令和检索结果的 .NET Framework 数据提供程序。ADO.NET 用于访问和操作数据的两个主要组件是.NET Framework 数据提供程序和 DataSet。ADO.NET 对数据的操作分为两种方式：连接方式和非连接方式。

.NET Framework 数据提供程序是专门为数据操作及快速、只进、只读访问数据而设计的组件。Connection 对象是提供到数据源的连接，使用 Command 对象可以访问用于返回数据、修改数据、运行存储过程及发送或检索参数信息的数据库命令，DataReader 对象可从数据源提供高性能的数据流。

DataAdapter 在 DataSet 对象和数据源之间起到桥梁作用。DataAdapter 使用 Command 对象在数据源中执行 SQL 命令以向 DataSet 中加载数据，并将对 DataSet 中数据的更改协调回数据源。

DataSet 是专门为独立于任何数据源的数据访问而设计的。因此，它可以用于多种不同的数据源，如 XML 数据，或管理应用程序本地的数据。DataSet 是包含一个或多个 DataTable 对象的集合，这些对象由数据行和数据列及有关 DataTable 对象中数据的主键、外键、约束和关系信息组成。有关 DataSet 使用会在后面的章节详细讲述。

如图 2-24 所示阐释了.NET Framework 数据提供程序和 DataSet 之间的关系。在图的左边是连接类，这些类实现直接与数据库通信、管理事务，以及从数据库检索数据和向数据库提交所做的更改；右边的类是非连接、允许用户脱机处理数据。

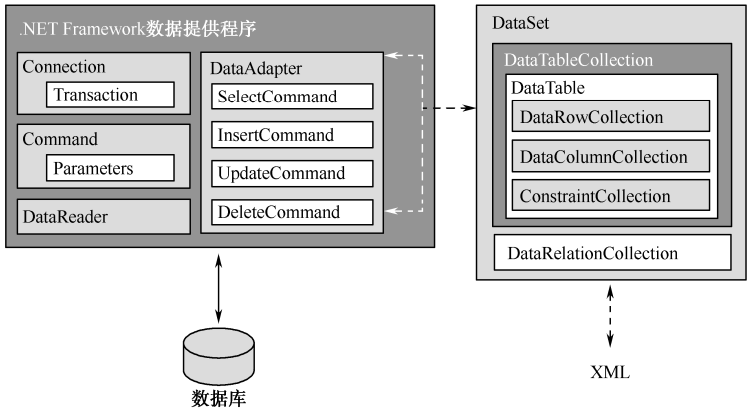

图 2-24 .NET Framework 数据提供程序和 DataSet 之间的关系

2. 命名空间

.NET 数据提供程序是一个类的集合，不同的数据源类型对应不同的.NET Framework 数据提供程序，.NET 数据提供程序都有自己的命名空间，不同数据源所对应的命名空间如下：

✧ System.Data.SqlClient 命名空间是 SQL Server 的.NET Framework 数据提供程序；

✧ System.Data.OracleClient 命名空间是 Oracle 的.NET Framework 数据提供程序；

✧ System.Data.Odbc 命名空间是 ODBC 的.NET Framework 数据提供程序；

✧ System.Data.OleDb 命名空间是 OLE DB 的 .NET Framework 数据提供程序。

用于连接方式的类根据连接数据源的不同在不同的命名空间中，如连接 SQL Server 要用 SqlConnection，在 System.Data.SqlClient 命名空间；连接 Oracle 数据库用 OracleConnection 类，在 System.Data.OracleClient 命名空间。而 System.Data 命名空间提供对表示 ADO.NET 结构类的访问。通过 ADO.NET 可以生成一些组件，用于有效管理多个数据源的数据。ADO.NET 结构的中心 DataSet 类在这个命名空间。

3. Connection 对象

用来创建与指定数据源的连接，创建连接实例代码如下（以 SQL Server 2005 为例）：

```
    string strconn= "server=(local)\SQLExpress;database=addressbook;user id=fyy;
Password=123"
    SqlConnection sqlconn=new SqlConnection( ConnectionString);
    Sqlconn.Open();
```

Connection 对象只提供了两个重载，其中 ConnectionString 是必须设置的属性，可以在实例化时赋初值，也可以在实例化后进行设置，代码如下：

```
    SqlConnection sqlconn=new SqlConnection();
    sqlconn.ConnectionString="server=(local)\SQLExpress;database=addressbook;
user id=fyy; Password=123"
```

其常用属性与方法如下：

✧ ConnectionString：获取或设置用于打开 SQL Server 数据库的字符串；

✧ ConnectionTimeout：获取在尝试建立连接时终止尝试并生成错误之前所等待的时间；

✧ Database：获取当前数据库或连接打开后要使用的数据库的名称；

✧ DataSource：获取要连接的 SQL Server 实例的名称；

✧ State：获取 SqlConnection 的状态；

✧ ClearAllPools()：清空连接池；

✧ ClearPool()：清空与指定连接关联的连接池；

✧ Close()：关闭与数据库的连接，这是关闭任何打开连接的首选方法；

✧ Open()：使用 ConnectionString 所指定的属性设置打开数据库连接。

注意：在完成与数据有关的操作后，数据库连接必须关闭。

4. 连接字符串

在创建连接实例时，不同数据源连接的不同之处只在于连接字符串，下面简单介绍其他两种常用数据源的连接字符串。

✧ Microsoft Office Access

其使用的标准代码如下：

```
    String strconn="Provider=Microsoft.Jet.OLEDB.4.0;Data Source=c:\bin\Local
Access40.mdb";
    OleDbConnection conn = new OleDbConnection(strconn);
```

✧ Oracle 数据库

其使用的标准代码如下：

```
    string strcon = "Data Source=Oracle8i;Integrated Security=yes";
    OracleConnection connection = new OracleConnection(strcon);
```

连接字符串是数据库连接的关键，下面把 SQL Server 2005 中常用的连接字符串涉及的关键字作简单介绍。

✧ Data Source|Server|Address：设置要连接的 SQL Server 实例的名称或网络地址；

✧ Initial Catalog|Database：设置数据库的名称；

✧ Integrated Security |Trusted_Connection：设置连接安全模式，当为 false 时，将在连接中指定用户 ID 和密码。当为 true 时，将使用当前的 Windows 账户凭据进行身份验证。可识别的值为 true、false、yes、no 及与 true 等效的 sspi（强烈推荐）；

✧ Password |Pwd：设置 SQL Server 账户登录的密码；

✧ User ID：设置 SQL Server 登录账户名；

✧ Connect Timeout：设置在终止尝试并产生错误之前，等待与服务器连接的时间长度（以秒为单位）。

说明：用“|”间隔的关键字表示所代表的意义相同。

注意：在 Web 应用程序中使用 Integrated Security 连接数据库时，在 SQL Server 数据库中需要添加 ASP.NET 的登录名，否则连接时会出现不允许远程连接的错误。

安全模式下本地 SQL Server 连接字符串代码如下：

```
    string strcon ="Data Source=.\SQLExpress;Initial Catalog=AdventureWorks;
Integrated Security=SSPI;";
```

5. 存储和访问数据库字符串

程序员在开发应用程序时，一般希望在不修改应用程序代码的情况下就能实现重新配置连接字符串，也就是实现连接配置从应用程序的其他部分分离出来。配置文件就是将这一信息从应用程序中分离出来的最好方法，可以将连接字符串和连接字符串的名字存储在配置文件中，并快速而容易地访问它们。设置和访问可参考实战演练。

ASP.NET 配置数据存储在命名为 Web.config 的 XML 文本文件中，Web.config 文件可以出现在 ASP.NET 应用程序的多个目录中。使用这些文件，可以在将应用程序部署到服务器之前、期间或之后方便地编辑配置数据。

ASP.NET 配置文件将应用程序配置设置与应用程序代码分开，可以方便地将设置与应用程序关联，在部署应用程序之后根据需要更改设置及扩展配置架构。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 9

拓展任务号	2-9	任务名称	Access 数据库的连接
计划用时	30 分钟	任务性质	课外
任务描述与目标			
在小型网站的制作中，经过会使用 Access 存储数据，通过本任务掌握到不同类型数据库的连接方式			
主要操作步骤提示			

1．把通讯录网站的 SQL Server 数据库导出为 Access 数据库文件；

2．打开 Exercise 网站，新建一个 Web 页面。打开 CS 代码文件，添加命名空间 using System.Data.OleDb；

3．添加一个按钮并添加按钮事件，事件参考代码如下：

```
protected void Button1_Click(object sender, EventArgs e)
{
  try
  {
   OleDbConnection oleconn = new OleDbConnection("Provider= Microsoft.Jet.OLEDB.4.0; Data Source=
data\address.mdb");
    oleconn.Open();
     Label1.Text = "数据库连接成功";
  }
  catch(Exception ex) Label1.Text=ex.Message;
}
```

2.4.2 创建 Command 数据操作

一、实战演练

（1）在“解决方案面板”中右击项目名，选择“添加”→“添加 ASP.NET 文件夹”→“App_Code”。

（2）右击“App_Code”，选择“添加”→“类”，把添加的类名命名为“StringControl”。在 StringControl 中添加四个方法，分别用于对用户输入的密码进行 MD5 加密、密码比较、判断用户是否输入了不必须输入的内容和提取文件的扩展名。如代码清单 2-16 所示。

代码清单 2-16 字符串加密

```
using System;
using System.Security.Cryptography;
using System.Text;
public class StringControl
{
    #region 获取字符串的 MD5 加密密文
    /// <summary>
    /// 获取字符串的 MD5 加密密文
    /// </summary>
    /// <param name="input">需要加密的字符串</param>
```

```
///<returns> 返回 string，MD5 加密密文</returns>
public static string GetMd5Hash(string input)
{
    MD5 md5Hash = MD5.Create();
    byte[] data = md5Hash.ComputeHash(Encoding.UTF8.GetBytes(input));
    StringBuilder sBuilder = new StringBuilder();
    for (int i = 0; i < data.Length; i++)
    {
        sBuilder.Append(data[i].ToString("x2"));
    }
    return sBuilder.ToString();
}
#endregion
#region 密文比对
/// <summary>
/// 密文比对
/// </summary>
/// <param name="fileName">文件名</param>
/// <param name="input">待比对的字符串</param>
/// <param name="hash">密文</param>
///<returns> 返回 bool,是否相等</returns>
public static bool VerifyMd5Hash(string input, string hash)
{
    MD5 md5Hash = MD5.Create();
    string hashOfInput = GetMd5Hash(input);
    StringComparer comparer = StringComparer.OrdinalIgnoreCase;
    if (0 == comparer.Compare(hashOfInput, hash))
    {
        return true;
    }
    else
    {
        return false;
    }
}
#endregion
# region 检查是不是空串，如果是，返回 NULL
/// <summary>
/// 检查是不是空串，如果是，返回 NULL
/// </summary>
/// <param name="str">待检查的字符串</param>
/// <returns>object</returns>
public static object isNull(string strValue)
{
    if (strValue != null)
    {
        if (strValue.Trim().Length == 0)
        {
            return DBNull.Value;
        }
        else
        {
            return strValue.Trim();
        }
    }
    else
        return DBNull.Value;
}
#endregion
```

```
        #region 提取扩展名
        /// <summary>
        /// 提取扩展名
        /// </summary>
        /// <param name="fileName">文件名</param>
        ///<returns> 返回 string，文件的扩展名</returns>
        public static string getExt(string fileName)
        {
            return fileName.Substring(fileName.LastIndexOf("."));
        }
        #endregion
    }
```

（3）打开“regist.aspx.cs”注册页面的代码文件。为注册用户添加变量定义 SqlCommand sqlcomm。

（4）删除 Page_Load 事件代码中的数据连接代码，双击“用户名”项后面的 Custom Validator 控件 CustomValidator1，在相应的事件代码中添加检测注册用户名是否存在，如代码清单 2-17 所示。

代码清单 2-17 CustomValidator1 自定义验证代码

```
    protected void CustomValidator1 (object source, ServerValidateEventArgs
args)
    {
    try
    {
       sqlconn = new SqlConnection
            (ConfigurationManager.ConnectionStrings["addressconn"] . ToString());
       sqlconn.Open();
    sqlcomm = new SqlCommand("select username from userinfo whereusername=@un",
sqlconn);
    sqlComm.Parameters.AddWithValue("@un", txtName.Text);
       if (sqlcomm.ExecuteScalar() == null)
              args.IsValid = true;
       else
              args.IsValid = false;
    }
    catch
    {
          Response.Redirect("error.htm");
    }
    finally
      {  if (sqlconn.State == Connection State.Open) dbase.closeconn(); }
    }
```

（5）回到“设计”视图，双击“确定”按钮，为完成注册功能添加代码，如代码清单 2-18 所示。

代码清单 2-18 注册功能代码

```
    protected void btnCertain_Click(object sender, EventArgs e)
    {
       if (Page.IsValid)
     {
        try
         {
           sqlConn = new SqlConnection(ConfigurationManager.ConnectionStrings
["addressconn"].ToString());
           sqlConn.Open();
```

```
          string filepath = Server.MapPath(@"~\");//"."也可以
          string fileFull, filename, hobby = "";
          if (filupImage.HasFile)
          {
            filename = @"pictures\" + DateTime.Now.Year.ToString() +
                    DateTime.Now.Month.ToString() + DateTime.Now.Day.ToString()
+
                 DateTime.Now.Hour.ToString() + DateTime.Now.Minute.ToString() +
                    DateTime.Now.Second.ToString() +

filupImage.FileName.Substring(filupImage.FileName.LastIndexOf('.'));
                    fileFull = filepath + @"\" + filename;
          }
          else
             filename = fileFull = "";
          for (int i = 0, j = 0; i < chkHobby.Items.Count; i++)
          {
              if (chkHobby.Items[i].Selected && j == 0)
                 { hobby = chkHobby.Items[i].ToString(); j++; }
                  else
                 { hobby = hobby + "," + chkHobby.Items[i]; j++; }
           }
          string commText = "insert into userinfo" +
                    "(username,userpassw,realname,sex,email,phone," +
                    "headpicture,datetimes,hobby,birthday,province)" +
                    "values(@un,@up,@rn,@sex,@em,@ph,@hp,@dt,@hb,@bt,@pr)";
         sqlComm=new SqlCommand(commText, sqlConn);
         sqlComm.Parameters.AddWithValue("@un", txtName.Text);
         sqlComm.Parameters.AddWithValue("@up",
StringControl.GetMd5Hash(txtPassword.Text));
         sqlComm.Parameters.AddWithValue("@rn",
StringControl.isNull(txtRname.Text));
        sqlComm.Parameters.AddWithValue("@sex", radSex.SelectedValue.ToString());
         sqlComm.Parameters.AddWithValue("@em", txtEmail.Text);
         sqlComm.Parameters.AddWithValue("@ph",
StringControl.isNull(txtPhone.Text));
         sqlComm.Parameters.AddWithValue("@hp",
StringControl.isNull(filename));
    sqlComm.Parameters.Add("@dt",SqlDbType.DateTime).Value  =   DateTime.Now.
ToShortDateString();
         sqlComm.Parameters.AddWithValue("@hb", StringControl.isNull(hobby));
         sqlComm.Parameters.Add("@bt",  SqlDbType.DateTime).Value  =  String
Control. isNull(txtbirth.Text);
         sqlComm.Parameters.AddWithValue("@pr",
StringControl.isNull(txtAddress.Text));
         if (sqlComm.ExecuteNonQuery() > 0)
          {
             filupImage.SaveAs(fileFull);
            Session["username"] = txtName.Text.Trim();
             Response.Redirect("~/cpmanage/cpmanage.aspx");
          }
           else
             {
              if (fileFull.Length != 0)
                {
                  FileInfo file1 = new FileInfo(fileFull);
                  if (file1.Exists) file1.Delete();
                }
              Response.Write("<script language=javascript>alert('注册不成功，请
重新注册');</script>");
```

```
            }
             }
             catch (Exception ex)
             {   throw (ex);         }
             finally { if (sqlConn.State == ConnectionState.Open) sqlConn.
Close(); }
          }
```

（6）启动调试，输入注册信息后，如果弹出“注册成功”对话框则说明注册成功。打开数据库中的用户信息表，查看信息是否注册成功。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-16 所示。

表 2-16 演练完成情况评价表

任 务 号	2-9	任 务 名 称	对数据库进行操作
任务子项	完成情况	主要问题	未完成原因
检测注册用户名是否存在的自定义事件			
注册用户事件			

三、知识点

Connection 对象用于建立与数据库连接的通道，Command 对象则是用于对数据库进行操作的类，通过 SQL 语句或存储过程实现操作。后面要介绍的 Datareader 和 DataAdapter 对象都要使用 Command 对象来从数据库中获取数据。

1. 创建 Command 对象

要使用 Command 对象，先进行实例化，Command 对象有四个构造函数重载。常用的 Command 对象实例代码如下（以 SQL Server 为例）：

```
SqlCommand sqlcomm = new SqlCommand();
sqlcomm.CommandText = "select * from userinfo";
sqlcomm.Connection = sqlconn;
sqlcomm.ExecuteScalar();
或 SqlCommand sqlcomm = new SqlCommand("select * from userinfo",sqlconn);
```

2. Command 常用的属性与方法

Command 对象的常用属性与方法如下：

✧ CommandText：获取或设置要对数据源执行的 SQL 语句或存储过程；

✧ CommandTimeout：获取或设置在终止对执行命令的尝试并生成错误之前的等待时间；

✧ CommandType：获取或设置一个指示如何解释 CommandText 属性的值；

✧ Connection：获取或设置此实例使用连接的 Connection 实例名；

✧ Parameters：获取 Command 对象 ParameterCollection；

✧ Cancel()：尝试取消 Command 的执行；

✧ CreateParameter()：创建 Parameter 对象的新实例；

✧ ExecuteXmlReader()：将 CommandText 发送到 Connection 并生成一个 XmlReader 对象。

在实例化 Command 对象时必须设置 CommandText 和 Connection 这两个属性，如果 CommandText 是存储过程，那么还必须设置 CommandType。CommandText 可以是 SQL 语句、

存储过程名或表名。

CommandType 是一个枚举，指定如何解释命令字符串。当 CommandType 属性设置为 StoredProcedure 时，CommandText 属性应设置为要访问存储过程的名称；当 CommandType 属性设置为 TableDirect 时，应将 CommandText 属性设置为要访问表的名称。默认值为 Text，CommandText 属性为 SQL 语句。

Parameters 是 SQL 语句或存储过程的参数集 ParameterCollection，默认为空集合。在实际应用中，如果 CommandText 属性有参数，就要通过为 Parameters 集的 Add()方法来添加元素及其值来完成。基本格式如下：

```
sqlcomm.Parameters.Add(参数名字符串，数据字段类型，[字段长度]).Value=值
```

Command 对象通过三个方法实现对数据库操作执行，这三个方法返回的结果不同。

✧ ExecuteNonQuery()方法：一般用于对数据表进行 UPDATE、DELETE 和 INSERT 时使用，通过返回的影响行数来判断是否实现对数据的操作。对于所有其他类型的语句，返回值为 -1。如果发生回滚，返回值也为-1。

✧ ExecuteReader()方法：通常与查询语句一起使用，并且返回一个数据阅读对象 SqlDataReader 类的一个实例。如果通过该方法执行更新语句，在命令成功执行的情况下也不会返回影响的行数。所以建议在对数据执行更新、插入、删除等操作时不用 ExecuteReader()方法。

✧ ExecuteScalar()方法：只用于检索数据库信息中的一个值，不需要返回表或影响行数等方面信息的情况，只需要返回结果的一些聚合函数 COUNT(*)、SUM()、AVG()等，这样 ExecuteScalar()方法就很有用。如果在一个常规的查询语句中调用该方法，则只读取第一行第一列，而丢弃其他所有的值。

3. ParameterCollection 常用的属性与方法

ParameterCollection 是 Command 相关联的参数的集合，以 MSSQL 的提供程序为例。SqlParameterCollection 类包含了以下常用属性和方法：

✧ Count：返回一个整数，包含在 SqlParameterCollection 中元素的数目；

✧ Item：获取具有指定属性的 SqlParameter，有两个获取格式 Item[Int32]和 Item[String]，可以通过下标方式或是参数名的方法访问。

✧Add()：此方法有六个重载，一个已经过时，常用的有三种重载方式：Add(SqlParameter)、Add(String, SqlDbType)、Add(String, SqlDbType, Int32)。

✧AddWithValue(): 将一个值添加到 SqlParameterCollection 的末尾，方法格式 AddWithValue(String，Object)，这个方法用于字符类型的数据处理比较好。

ParameterCollection 属于一种集合，基本集合的方法基本适用。可以通过 Clear()、Remove()对成员进行处理。

在实际过程中使用 Add()与 AddWithValue()那两个方法更好，没有一定的定论，AddWithValue()需要猜测参数的数据类型，有时会出现一些意外，但是书写更简洁。

四、任务拓展

本节完成三个拓展实践任务，两个课内实践任务，一个课外实践任务。

拓展任务卡 10

拓展任务号	2-10	任务名称	完成登录功能
计划用时	15 分钟	任务性质	课内

任务描述与目标

在网络通讯录中，联系人的管理只能在用户登录后才能进行，用户登录功能是通过用户输入的用户名和密码与数据库中的用户信息进行比较实现的

主要操作步骤提示

1．打开通讯录网站中的 login.ascx 用户控件；

2．添加设置连接字符串的属性，参考代码如下：

```
string connstr= ConfigurationManager.ConnectionStrings["addressconn"] . ToString();
public string LoginConnStr
{
    set { connstr = value; }
    get { return connstr; }
}
```

3．选择“登录”按钮并添加按钮事件，事件参考代码如下：

```
protected void ImageButton1_Click(object sender, ImageClickEventArgs e)
{
 try
 {
   SqlConnection sqlconn = new SqlConnection(connstr);
   sqlconn.Open();
   SqlCommand sqlcomm = new SqlCommand("select username from userinfo where username=@un and
userpassw=@upw", sqlconn);
   sqlcomm.Parameters.Add("@un", SqlDbType.VarChar).Value = txtname.Text;
   sqlcomm.Parameters.Add("@upw", SqlDbType.VarChar).Value = StringControl.GetMd5Hash(txt
Password.Text);
   if (sqlcomm.ExecuteScalar() != null)
   {
    Session["username"] = txtname.Text.Trim();
    Page.Header.Title = txtname.Text + ",欢迎您回来! ";
    Response.Redirect("contactormanage.aspx");
   }
   else
   Response.Write("<script  language=javascript>alert('用户名不存在或密码错，请重新登录
');</script>");
  }
 catch (Exception ex)
  {
   Response.Redirect("error.html");
   }
  finally
   { if (sqlconn.State == ConnectionState.Open) dbase.closeconn(); }
}
```

4．打开 Default.aspx 文件，重新拖入登录用户控件，并对用户控件的属性进行设置

拓展任务卡 11

拓展任务号	2-11	任务名称	完成添加联系人功能
计划用时	20 分钟	任务性质	课内
任务描述与目标			
在网络通讯录中，注册页面的功能与添加联系人功能相似，通过添加联系人页面练习进一步掌握 Command 对象的属性与方法			
主要操作步骤提示			
1．打开 addcontactor.aspx 页面； 2．添加 Page_Load 事件，事件代码参考管理联系人页面的 Page_Load 事件； 3．双击“确定”按钮并添加按钮事件，事件代码参考注册页面代码			

拓展任务卡 12

拓展任务号	2-12	任务名称	在显示详细信息页面中添加删除信息功能
计划用时	30 分钟	任务性质	课外
任务描述与目标			
要删除和更新联系人信息，应先查找到联系人的信息，通过在查找页面中单击联系人，并把联系人的 ID 号通过参数传递到本页面			
主要操作步骤提示			

1．打开 detailinfo.aspx 页面；

2．添加 Page_Load 事件，事件参考代码如下：

```
protected void Page_Load(object sender, EventArgs e)
{
if (!Page.IsPostBack)
    {   id = Convert.ToInt32(Request.QueryString["cid"]);
     Session["id"] = id.ToString();
   }
}
```

3．双击“删除联系人”按钮并添加按钮事件，事件参考代码如下：

```
protected void ButDele_Click(object sender, EventArgs e)
{
  SqlConnection  sqlconn=new  SqlConnection(ConfigurationManager.ConnectionStrings["addressconn"].
ToString());
    sqlconn.Open();
     int id1 = Convert.ToInt32(Session["id"]);
    SqlCommand sqlcomm = new SqlCommand("delete  from contactinfo where conID=@cid", sqlconn);
     sqlcomm.Parameters.Add("@cid", SqlDbType.Int).Value = id;
     if( sqlcomm.ExecuteNonQuery()>0)
       Response.Write("<script language=javascript>alert('删除成功');</script>");
}
```

2.4.3 DataReader 数据对象

一、实战演练

（1）分析查询联系人功能，不同的用户只能查询到自己的联系人，登录用户的信息通过会话变量来获取。

（2）在 cpmanage 文件夹中打开查找 fingcp.aspx 联系人页面。切换到“设计”视图，选择

“家人”的“LinkButton”，在“事件”面板中，双击 Click 事件。

（3）选择其他几个类型的“LinkButton”，把它们的 Click 事件关联到“家人”的 Click 事件。

（4）转换到“源”视图，添加“LinkButton”的 Click 事件代码，如代码清单 2-19 所示。

代码清单 2-19 分类查找“LinkButton”的 Click 事件代码

```
    protected void LinkButton_Click(object sender, EventArgs e)
        {
           LinkButton linkbtn = (LinkButton)sender;
            string Classinfo = linkbtn.Text;
            if (Session["username"] != null)
            {
               try
                {
                    sdr = findclass(Classinfo, Session["username"].ToString());
//获取查询结果
                    gettable(sdr);    //生成显示内容的方法
                    sdr.Close();
                }
                catch
                { Response.Redirect("error.htm");  }
    finally
                { if (sqlConn.State == ConnectionState.Open) sqlConn.Close(); }
            }
        }
```

（5）添加 findclass ()方法实现信息查询，如代码清单 2-20 所示。

代码清单 2-20 findclass ()方法代码

```
    SqlDataReader findclass(string fc,string uname)
    {
    SqlDataReader sqldr=null;
         try
         {
           sqlConn = new SqlConnection(ConfigurationManager.ConnectionStrings
["addressconn"].ToString());
           sqlConn.Open();
           sqlComm = new SqlCommand();
           sqlComm.CommandText = "select conID,cname as 姓名,homeaddress as 家庭
地址,"+
    "company as 公司名 from contactinfo where username=@un and relation=@rl";
           sqlComm.Connection = sqlConn;
           sqlComm.Parameters.Add("@un", SqlDbType.VarChar).Value = uname;
           sqlComm.Parameters.Add("@rl", SqlDbType.VarChar).Value = fc;
           sqldr = sqlComm.ExecuteReader();
          }
           catch { }
           return sqldr;
       }
```

（6）添加 gettable()实现在页面内以表格的形式显示查询结果，如代码清单 2-21 所示。

代码清单 2-21 gettable()方法代码

```
    void gettable(SqlDataReader sqldr)
    {
         if (sdr.Read())
         {
           HtmlTable tabl1 = new HtmlTable();
```

```
            tabl1.ID = "t1";
            tabl1.Width = "360";
            tabl1.Border = 1;
            tabl1.BorderColor = "#092a49";
            HtmlTableRow row= new HtmlTableRow();
            HtmlTableCell cell;
            //读取数据表字段信息并显示表头
            for (int cellcount = 1; cellcount < sdr.FieldCount; cellcount++)
            {
                cell = new HtmlTableCell("th");
                cell.Controls.Add(new
LiteralControl(sdr.GetName(cellcount).ToString()));
                row.Cells.Add(cell);
            }
            tabl1.Rows.Add(row);
            //读取数据表中的每行信息并显示在表中
        do
            {
              row = new HtmlTableRow();
              for (int cellcount = 1; cellcount < sdr.FieldCount; cellcount++)
              {
                cell = new HtmlTableCell();
                if (cellcount == 1)
                {  cell.Controls.Add(new  LiteralControl("<a   href='detailinfo.
aspx?cid=" +
                                sdr.GetValue(cellcount - 1).ToString() + "'>" +
                                sdr.GetValue(cellcount).ToString() + "</a>"));
                        }
               else
                   cell.Controls.Add(new
LiteralControl(sdr.GetValue(cellcount).ToString()));
                row.Cells.Add(cell);
               }
              tabl1.Rows.Add(row);
            } while (sdr.Read());
            Panel1.Controls.Clear();
             Panel1.Controls.Add(tabl1);
        }
         else
            Panel1.Controls.Add(new LiteralControl("这个类别没有联系人"));
    }
```

（7）运行调试，单击“联系人类别”按钮，查看运行结果。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-17 所示。

表 2-17　演练完成情况评价表

任　务　号	2-10	任 务 名 称	实现分类查找联系人功能
任务子项	完成情况	主要问题	未完成原因
添加 findclass ()方法			
添加 gettable()			
添加 LinkButton 的 Click 事件			
关联所有的联系人类型事件			

三、知识点

在连接模式中，Command 对象将执行 ExecuteReader()方法后的结果返回给 DataReader 对象，通过 DataReader 对象进行数据处理。DataReader 对象只允许以只读、顺向的方式查看其中所存储的数据，并在 ExecuteReader()方法执行期间进行实例化。DataReader 对象无论是在系统开销还是性能方面都比较有效，它在任何时候都只缓存一个记录，并且没有把整个结果载入内存的等待时间，从而避免了使用大量内存，提高了性能。

1. DataReader 类简述

DataReader 从数据库中检索只读、只进的数据流。查询结果在查询执行时返回，并存储在客户端的网络缓冲区中，直到使用 DataReader 的 Read 方法时对它们发出请求。

DataReader 不能直接创建对象的实例，而是通过调用 Command.ExecuteReader 创建。

每次使用完 DataReader 对象后都应调用 Close 方法将其关闭。如果 Command 包含输出参数或返回值，那么在关闭 DataReader 之前，无法访问这些输出参数或返回值。注意，当 DataReader 打开时，该 DataReader 以独占方式使用 Connection；在 DataReader 关闭之后，将无法对 Connection 执行任何命令，包括创建另一个 DataReader。

每个.NET Framework 数据提供程序都包括一个 DataReader 对象，即用于 OLEDB 的.NET Framework 数据提供程序包括 OleDbDataReader 对象，用于 SQL Server 的.NET Framework 数据提供程序包括 SqlDataReader 对象；用于 ODBC 的.NET Framework 数据提供程序包括 OdbcDataReader 对象；用于 Oracle 的.NET Framework 数据提供程序包括 OracleDataReader 对象。

2. DataReader 类的属性和方法

以 SqlDataReader 类为例，DataReader 类常用的属性和方法如下：

✧ Connection：获取与 SqlDataReader 关联的 SqlConnection；

✧ Depth：获取一个值，用于指示当前行的嵌套深度；

✧ FieldCount：获取当前行中的列数，如果没有有效记录集则为 0，否则为当前行中的列数；

✧ HasRows：获取一个值，用于指示 SqlDataReader 是否包含一行或多行，包含一行或多行为 true，否则为 false；

✧ IsClosed：检索一个布尔值，用于指示是否已关闭指定的 SqlDataReader 实例；

✧ Item：索引器属性，以原始格式获得一列的值，可以通过 Item[Int32]和列名称 Item[Stmg]来读取数据行的字段值；

✧ RecordsAffected：获取执行 SQL 语句所更改、插入或删除的行数；

✧ Close()：关闭 SqlDataReader 对象；

✧ GetFieldType()：获取对象数据类型的 Type；

✧ GetName()：获取指定列的名称；

✧ GetSchemaTable()：返回一个 DataTable，它描述 SqlDataReader 列的元数据；

✧ GetValue()：获取以本机格式表示的指定列的值；

✧ GetValues()：获取当前行的集合中的所有属性列；

✧ IsDBNull()：获取一个值，用于指示列中是否包含不存在的或缺少的值；

✧ NextResult()：当读取批处理 Transact-SQL 语句的结果时，使数据读取器前进到下一个结果；

✧ Get<DataType>()：根据字段的序号，以指定的类型返回当前行内字段的内容。

✧ Read()：使 SqlDataReader 前进到下一条记录；

Read()方法在调用 Read()方法之前 SqlDataReader 的默认位置在第一条记录前面，因此必须调用 Read()才能开始访问数据，如果存在则返回值为 true；否则为 false。通过判断 Read()的返回值可以指示查询是否还有存在数据。GetValue(int i)方法的使用方式类似 Item 属性，返回值为 Object 类型的字段内容。Get<DataType>(int i)系列方法仅接受整数作为字段索引器，因此在获取数据值时用 Get<DataType>方法比用 GetValue()的性能更好。代码示例如下：

```
sqldr = sqlcomm.ExecuteReader();
       string strname;
       strname = sqldr.GetString(1);    //这种方式更好
       //or
   strname = Convert.ToString(sqldr.GetValue(1));
```

GetValues()方法则把数据行存储在一个 Object 类型的数组中。如果想事先获取数据，性能上优于以每列的方式读取。代码示例如下：

```
sqldr = sqlcomm.ExecuteReader();
object[] Odata = new object(sqldr.FieldCount);
sqldr.GetValues(Odata);
string strdata="";
foreach (object x in Odata)
{
   strdata = strdata + "," + x.ToString();
}
Response.Write(strdata);
```

NextResult()当处理返回多个结果的批量查询时，可以使用该方法移动到下一个结果，与 Read()方法类似，如果存在结果则为 true；否则为 false。代码示例如下：

```
SqlCommand command = new SqlCommand(
        "SELECT CategoryID, CategoryName FROM dbo.Categories;" +
        "SELECT EmployeeID, LastName FROM dbo.Employees",
        connection);
SqlDataReader reader = command.ExecuteReader();
while (reader.HasRows)
{
    while (reader.Read())
          {
           Response.Write(rereader.GetInt32(0)+ reader.GetString(1));
          }
    reader.NextResult();
  }
```

四、任务拓展

本节完成两个拓展实践任务，一个课内实践任务，一个课外实践任务。

拓展任务卡 13

拓展任务号	2-13	任务名称	条件查询
计划用时	15 分钟	任务性质	课内

任务描述与目标

除了分类查询外，信息查询应该还能实现条件查询

主要操作步骤提示

1．打开查找 findlinker.aspx 联系人页面；

2．双击“查找”按钮并添加按钮事件，事件参考代码如下：

```
protected void Button1_Click(object sender, EventArgs e)
    {
      string sqlstr;
      if (txtkey.Text != "")
      {
        switch (DropDownList1.Text)
        {
          case "姓名":
          sqlstr = "select conid, cname as 姓名,homeaddress as 家庭地址,comaddress as 公司地址 from
contactinfo where cname like @key and username=@un";
          sdr = findkey(sqlstr, txtkey.Text, Session["username"].ToString());
          gettable( sdr);
          break;
          case "家庭地址":
          sqlstr = "select conid, cname as 姓名,homeaddress as 家庭地址,comaddress as 公司地址 from
contactinfo where homeaddress like @key and username=@un";
          sdr = findkey(sqlstr, txtkey.Text, Session["username"].ToString());
          gettable(sdr);
          break;
          case "公司地址":
          sqlstr = "select conid, cname as 姓名,homeaddress as 家庭地址,comaddress as 公司地址 from
contactinfo where comaddress like @key and username=@un";
          sdr = findkey(sqlstr, txtkey.Text, Session["username"].ToString());
          gettable(sdr);
          break;
       }
     }
}
```

3．添加 findkey()方法，代码参考 findclass ()方法，只是参数多了一个字符串

拓展任务卡 14

拓展任务号	2-14	任务名称	在显示详细信息页面中添加、删除和更新信息功能
计划用时	30 分钟	任务性质	课外

任务描述与目标

要删除和更新联系人信息，应先查找到联系人的信息，通过在查找页面中单击联系人，并把联系人的 ID 通过参数传递到本页面

主要操作步骤提示

1．打开 detailinfo.aspx 页面；

2．添加 Page_Load 事件，事件参考代码如下：

```
    protected void Page_Load(object sender, EventArgs e)
    {
      if (!Page.IsPostBack)
        {   id = Convert.ToInt32(Request.QueryString["cid"]);
```

续表

拓展任务号	2-14	任务名称	在显示详细信息页面中添加、删除和更新信息功能
计划用时	30 分钟	任务性质	课外

```
        Session["id"] = id.ToString();
        Showinfo(id);
      }
    }
```

3．Showinfo(id)方法的主要参考代码如下：

```
    void Showinfo(int id)
    {
      SqlConnection sqlconn=new SqlConnection(ConfigurationManager.ConnectionStrings["addressconn"].
ToString());
       sqlconn.Open();
       SqlCommand sqlcomm = new SqlCommand("select * from contactinfo where conID=@cid",
sqlconn);
           sqlcomm.Parameters.Add("@cid", SqlDbType.Int).Value = id;
           SqlDataReader sdr = sqlcomm.ExecuteReader();
           if (sdr.Read())
           {
              Lblname.Text=sdr.GetValue(1).ToString();
              Txtname.Text = Lblname.Text;
              if (sdr.GetValue(2).ToString() == "男")
              {
                 RadioSex2.Items[0].Selected = true;
                 RadioSex1.Items[0].Selected = true;
              }
              else
              {
                 RadioSex2.Items[1].Selected = true;
                 RadioSex1.Items[1].Selected = true;
              }
               LblHtele.Text=sdr.GetValue(3).ToString();
               TxtHomeTel.Text = LblHtele.Text;
               LblHaddress.Text = sdr.GetValue(4).ToString();
               TxtHomeAddress .Text= LblHaddress.Text;
              LblMobile.Text=sdr.GetValue(5).ToString();
              Txtmobile.Text=LblMobile.Text; lblcompany.Text=sdr.GetValue(6).ToString();
              Txtcompany.Text = lblcompany.Text;
              LblCtele.Text=sdr.GetValue(7).ToString();
              TxtCompanyTel.Text = LblCtele.Text;
              Lbladdress.Text=sdr.GetValue(8).ToString();
              TxtCompanyAddress.Text = Lbladdress.Text;
              Lblcon.Text = sdr.GetValue(9).ToString();
              for (int i = 0; i < DropDownRelation.Items.Count; i++)
              {
                if (DropDownRelation.Items[i].Text.Trim() == Lblcon.Text.Trim())
                DropDownRelation.Items[i].Selected = true;
              }
           }
           sdr.Close();
           sqlconn.Close();
    }
```

4．双击“更新”按钮并添加按钮事件，事件参考代码如下：

```
 protected void ButUpdate_Click(object sender, EventArgs e)
     {
      SqlConnection sqlconn = new SqlConnection(ConfigurationManager.ConnectionStrings["addressconn"].
ToString());
```

续表

拓展任务号	2-14	任务名称	在显示详细信息页面中添加、删除和更新信息功能
计划用时	30 分钟	任务性质	课外

```
        sqlconn.Open();
        SqlCommand sqlcomm = new SqlCommand("update contactinfo set cname=@cn,sex=@sx,homephone=@hp,
homeaddress=   @hadd,mobphone=@mp,company=@company,comphone=@cp,comaddress=@cadd,relation=@rel
where conID=@id", sqlconn);
        sqlcomm.CommandType = CommandType.Text;
        sqlcomm.Parameters.Add("@cn", SqlDbType.NVarChar).Value = Txtname.Text;
        sqlcomm.Parameters.Add("@sx", SqlDbType.NChar).Value = RadioSex2.SelectedItem.Text;
        sqlcomm.Parameters.Add("@hp", SqlDbType.VarChar).Value = TxtHomeTel.Text;
       sqlcomm.Parameters.Add("@hadd", SqlDbType.NVarChar).Value = TxtHomeAddress.Text;
        sqlcomm.Parameters.Add("@mp", SqlDbType.Char).Value = Txtmobile.Text;
        sqlcomm.Parameters.Add("@company", SqlDbType.NVarChar).Value =Txtcompany.Text;
        sqlcomm.Parameters.Add("@cp", SqlDbType.Char).Value = TxtCompanyTel.Text;
        sqlcomm.Parameters.Add("@cadd", SqlDbType.NVarChar).Value = TxtCompanyAddress.Text;
        sqlcomm.Parameters.Add("@rel", SqlDbType.NChar).Value =DropDownRelation.SelectedItem.
Text;
        int id=Convert.ToInt32( Session["id"]);
        sqlcomm.Parameters.Add("@id", SqlDbType.Int).Value =id;
        if(sqlcomm.ExecuteNonQuery()>0)
           Response.Write("<script language=javascript>alert('更新成功');</script>");
  }
```

5. 添加“显示信息”按钮事件，事件参考代码如下：

```
  protected void showinfomation_Click(object sender, EventArgs e)
     {   Panel1.Visible = true;
         Panel2.Visible = false;
        Showinfo(Convert.ToInt32(Session["id"]));
     }
```

6. 添加“修改信息”按钮事件，事件参考代码如下：

```
  protected void upDateInfomation_Click(object sender, EventArgs e)
     {
         Panel1.Visible = false;
         Panel2.Visible = true;
     }
```

2.5 任务 3 网站的调试与发布

在编写程序过程中，难免会遇到一些错误，为了解决这些错误，需要对应用程序进行调试，查出错误并进行处理。网站开发完成后，要把网站发布到服务器上。

解决方案

为完成本任务，要求完成以下几个方面的工作：

（1）对网站各个页面进行调试，排除单页面的逻辑错误；

（2）能准确使用 Visual Studio 的调试工具和调试技术；

（3）能对网站进行整体测试与排错；

（4）能对网站发布、打包和安装。

2.5.1 .NET 平台的调试工具

Visual Studio 2012 提供了一个强大的调试工具，用户可以通过它观察程序运行时的行为并确定逻辑错误的位置。

一、实战演练

（1）打开网络通讯录应用程序，启用调试功能。要对 Web 应用程序进行调试，必须通过项目属性页中的启用调试器。在“解决方案资源管理器”中右击项目名称，在弹出的快捷菜单中选择“属性页”选项，在左边的选择项中选择“启动选项”，打开如图 2-25 所示的“属性页”对话框。

（2）在“属性页”对话框中选择“ASP.NET”复选框。单击“应用”按钮，完成调试器的启动，其他选择项的设置如图 2-25 所示。一般来说，Web 应用程序的默认状态为启动状态。

（3）显示“调试”工具栏，调试工具栏的默认状态为显示状态，如果没有显示，可以通过设置来启动。在菜单栏执行“工具”→“自定义”命令，在弹出如图 2-26 所示的“自定义”对话框中选择“调试”复选框。在工具栏上方就可以看到如图 2-27 所示的“调试”工具栏。

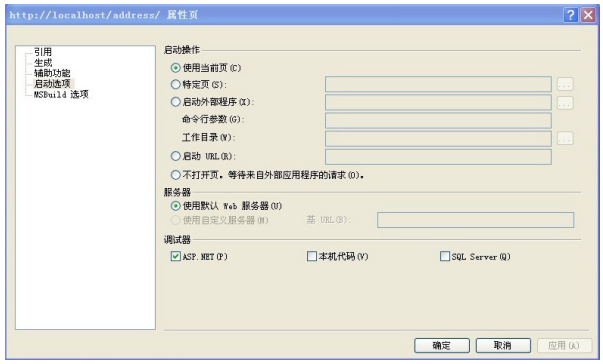

图 2-25 “属性页”对话框

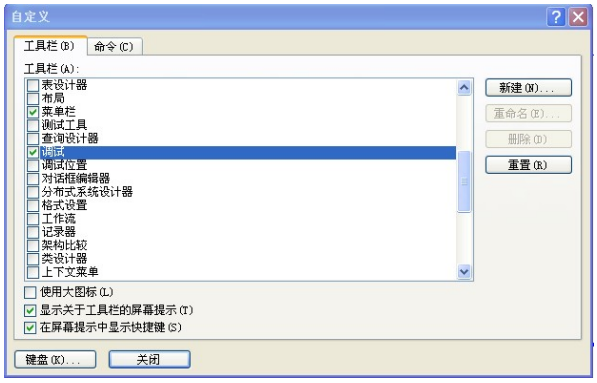

图 2-26 “自定义”对话框

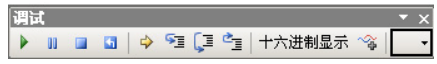

图 2-27 “调试”工具栏

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-18 所示。

表 2-18 演练完成情况评价表

任 务 号	2-11	任 务 名 称	.NET 平台的调试工具配置
任务子项	完成情况	主要问题	未完成原因
属性页配置			
添加调试工具			

三、知识点

应用程序代码可能会包含各种类型的错误或 Bug，在编译过程中将捕获大部分语法错误。但是，其他类型的错误需要调试代码，即在代码运行时检查代码以验证其执行路径和数据是否正确。

若要启用调试，必须在“项目属性”页和应用程序的 web.config 文件中都启用。在 web.config 文件中找到<compilation>标记，将 debug 设置为 true 属性。注意，属性是区分大小写的，因此请确保指定的是“debug”，而不是“Debug”或“DEBUG”。设置后的代码如下：

```
<compilation debug="true">
        ……
</compilation>
```

“调试”工具栏中的前四个按钮是可以手动的。第一个是“启动调试”按钮，该按钮与“标准”工具栏中的“启动调试”按钮功能相同；另外三个呈灰色的按钮只有在程序启动后才能正常使用。“调试”工具栏中的常用按钮功能说明如表 2-19 所示。

表 2-19 “调试”工具栏中的常用按钮功能说明

按 钮	图 标 文 本	快 捷 键	说 明
	启动调试	【F5】	在程序执行过程中启动或继续
	全部中断	【Ctrl+Alt+Break】	停止当前正在运行的应用程序
	停止调试	【Shift+F5】	停止调试
	重新启动	【Shift+Ctrl+F5】	停止当前正在调试的应用程序，然后重新启动
	显示下语句	【Alt+数字键*】	显示下几个语句
	逐语句	【F11】	进行单步执行当前语句
	逐过程	【F10】	如果当前行包括一个方法或函数的调用，则不会逐语句执行当前行中的方法或函数，而是执行到调用的一行
	跳出	【Shift+F11】	如果当前行在一个方法或函数内，则执行完该方法或函数后，调试器停止在调用行之后这一行
	即时	【Ctrl+D,I】	可以选择调试窗口

四、任务拓展

本节完成两个课内拓展实践任务。

拓展任务卡 15

拓展任务号	2-15	任务名称	调试设置练习
计划用时	5 分钟	任务性质	课内
任务描述与目标			
利用 Web.config 配置调试属性，学生经过自主练习进一步认识 Web.config 文件的作用			
主要操作步骤提示			
1．打开任务的“属性页”，取消对“ASP.NET”复选框前的选择，打开通讯录中的任意一个 CS 文件，在代码中任意处插入一个断点； 2．打开 Web.config 文件，设置<compilation> 节的属性 debug="true"，运行程序，查看运行情况； 3．设置<compilation> 节的属性 debug="false"，在“属性页”选择 “ASP.NET”复选框前，运行程序，查看运行情况			

拓展任务卡 16

拓展任务号	2-16	任务名称	调试工具栏中各按钮的使用
计划用时	10 分种	任务性质	课内
任务描述与目标			
1．练习调试工具栏按钮的使用； 2．观察调试工具的可用情况			
主要操作步骤提示			
1．添加调试工具栏； 2．选择打开一个 Web 窗体文件，分别单击  按钮启动运行，在浏览器窗口中观察页面的事件运行情况； 3．单击  按钮，在浏览器窗口中观察页面的事件运行情况； 4．分别单击  按钮，观察运行情况； 5．单击  按钮，停止程序运行，在代码中设置断点后运行程序，观察调试工具栏前后的变化； 6．逐一使用后面的调试按钮，观察代码的运行情况			

2.5.2 代码跟踪

开发项目发生错误是难免的，一些语法性的错误比较容易解决，但逻辑错误解决起来比较困难。本节主要学习如何排查逻辑错误。

一、实战演练

（1）打开项目中的“regist.aspx.cs”文件，把鼠标指针定位在要进行调试的代码处右击，在弹出的快捷菜单中执行“断点”→“插入断点”命令。插入断点后的效果如图 2-28 所示。

（2）运行程序，程序会在断点处停止，同时该代码行会变成黄色。通过按【F11】键可以向前移动一行。当鼠标移到断点处的变量时，可以看到如图 2-29 所示变量值的变化。

（3）在排查逻辑错误时往往要查看某些变量值的变化。对于简单的变量，鼠标移动到变量时就会自动显示变量的值。对于数组变量、类变量等则在下拉菜单中执行“调度”→“快速监视”命令，使用如图 2-30 所示的“快速监视”对话框来查看变量的值更适合。

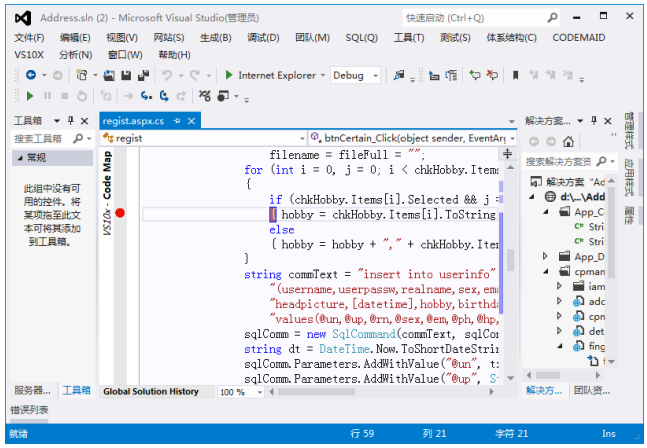

图 2-28　插入断点后的效果

```
for (int i = 0,j=0; i < chkHobby.Items.Count; i++)
{

    if (chkHobby.Items[i].Selected && j == 0)
    {
        hobby = chkHobby.Items[i].ToString();
        j++;
    }
    else
    { hobby = hobby + "," + chkHobby.Items[i]; j++; }
}
```

图 2-29　变量值的变化

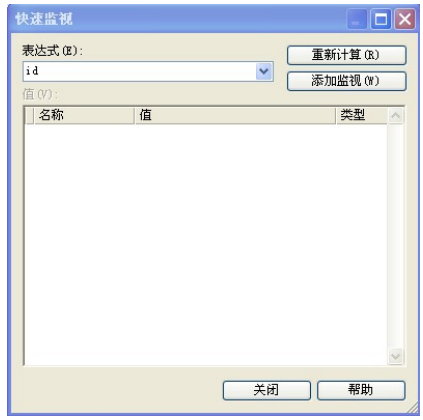

图 2-30　“快速监视”对话框

（4）在“快速监视”对话框中输入表达式，单击“添加监视”按钮，打开如图 2-31 所示“监视”对话框。关闭“快速监视”对话框，按【F11】键逐语句执行，可以通过变量值的变化查看。

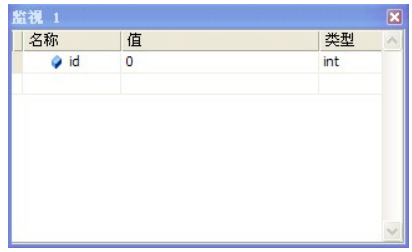

图 2-31　“监视”对话框

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-20 所示。

表 2-20　演练完成情况评价表

任　务　号	2-12	任 务 名 称	代码调试技术
任务子项	完成情况	主要问题	未完成原因
断点设置			
添加快速监视			
逐语句运行			

三、知识点

Visual Studio 调试器提供了强大的调试功能，可以观察程序运行时的行为并确定逻辑错误的位置。调试器可用于所有的 Visual Studio 编程语言及其关联的库。使用调试器，可以中断（或挂起）程序的执行以检查代码、计算和编辑程序中的变量、查看寄存器、查看从源代码创建的指令，以及查看应用程序所占用的内存空间。使用“编辑并继续”功能，可以在调试时对代码进行更改，然后继续执行。

1. 设置断点

断点是一个信号，它通知调试器在某个特定点上暂时将程序挂起。当执行到某个断点时，程序处于中断模式，进入中断模式不会终止或结束程序的执行，执行可以在任何时候恢复。程序员可以在需要时在中断模式中对程序进行调整。断点调试与使用逐语句检查代码不同的是，可以让程序一直执行，直到遇到断点才开始调试，大大加快了调试的速度。没有这个功能，调试较大的程序几乎不可能。

设置断点最快的方式是，打开要进行调试的代码文件，把鼠标移动到要设置断点的语句左边代码编辑器外面的灰色地带，当鼠标指针变成指向左边的箭头时单击，就可以设置断点。设置断点也可以右击代码，在弹出的快捷菜单中执行“断点”→“插入断点”命令。

通过断点窗口可以查看断点信息，打开断点窗口有三种方式：执行“调试”→“窗口”→“断点”命令；在“调试”工具栏中单击按钮，在下拉列表中选择“断点”项；按【Ctrl+Alt+B】组合键。

右击断点左边的红色圆球，通过设置弹出的如图 2-32 所示快捷菜单中的各个命令可对断点的属性进行设置。

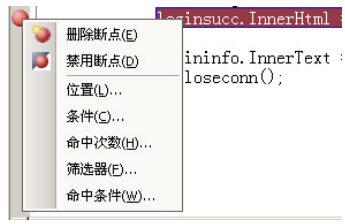

图 2-32　断点属性快捷菜单

2. 单步调试代码

设置好断点后，结合单步调试，可以检查执行流程和测试变量的值、属性、对象等信息，较好地完成程序调试。一般在程序运行到断点处时，按【F11】键向前移动语句执行。

3. 跟踪点

跟踪点是 Visual Studio 中新的调试器功能。跟踪点是带有与断点关联的自定义操作的断点。选中跟踪点时，调试器会执行指定的跟踪点操作，而不是或不仅是中断程序执行。跟踪点常用于在程序到达某点时输出消息。使用跟踪点可以实现许多原本要使用 Trace 工具才能实现的功能，并且使用跟踪点无需修改代码。另一个不同之处在于只有在调试器工作时，跟踪点才可以起作用。

可以用下面两种不同的方法来创建跟踪点：

✧ 通过添加跟踪点操作将现有断点转换为跟踪点，任何类型的断点都可以转换为跟踪点；

✧ 使用“新建跟踪点”命令从头开始创建跟踪点。

4. 异常处理

在开发 Web 应用程序时出现错误是难免的，应该未雨绸缪，为可能出现的错误提供恰当的处理。

使用 try……catch……fanlly 语句来捕捉各种异常是常用的一种方法，例如：

```
try
    {
        ……
```

```
        }
        Catch(Exception ex)
        {
            Response.Write(ex.Message);
        }
```

在 ASP.NET 中还可以通过设置 Web.config 配置文件在应用程序范围内实现自定义的错误处理页面。错误处理配置信息在 Web.config 配置文件的<customErrors>节中，基本配置结构代码如下：

```
<configuration>
   <configSections>
     ……
   </configSections>
   <appSettings/>
    <connectionStrings/>
   <system.web>
       ……
     <customErrors defaultRedirect="error.html" mode="On">
     </customErrors>
   </system.web>
</configuration>
```

2.5.3 发布与部署

网站项目开发完成后，需要对网站整体测试，选择合适的方式再进行部署。

一、实战演练

（1）打开通讯录项目，在“解决方案资源管理器”面板中，右击登录页“Default.aspx”，在弹出的快捷菜单中选择“设为起始页”选项。

（2）启动项目调试，从“注册用户”开始，模拟用户对所有功能进行测试。同时查看数据库中数据的变化是否符合功能要求。测试无误后对网站进行打包。

（3）在“解决方案资源管理器”面板中，右击解决方案，在弹出的快捷菜单中执行“添加”→“新建项目”命令。打开“添加新项目”对话框，如图 2-33 所示。在“项目类型”列表中选择“其他项目类型”下的“安装和部署”，在“模板”中选择“启用…… 安装和部署”，在“名称”和“位置”中分别输入合适的值。

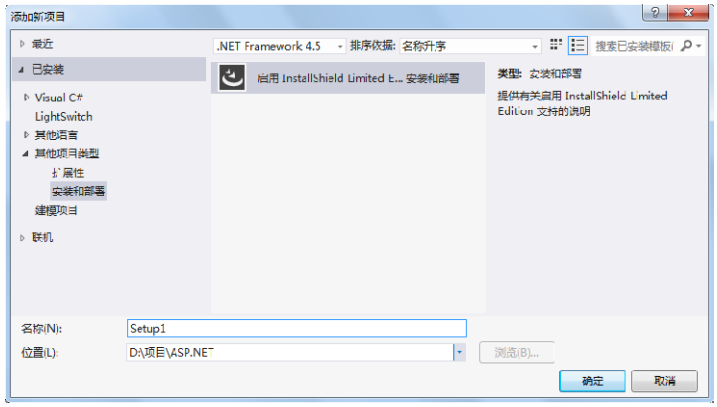

图 2-33 “添加新项目”对话框

（4）新建一个项目，单击“确定”按钮，如果是第一次使用的话，会打开如图 2-34 所示的 InstallShield 下载提示网页。

图 2-34　InstallShield 下载网页界面

（5）选择“步骤 2：转到下载网站”，在如图 2-35 所示的 InstallShield 下载信息填写页面，填写完右边的信息，单击“download now”即会收到一封邮件，里面有下载地址和激活码，同时页面也会跳转到如图 2-36 所示的激活码显示及下载页面。单击上面的链接进行下载，会下载到一个名为：InstallShield2013LimitedEdition.exe 的安装包，用管理员权限运行安装完成。

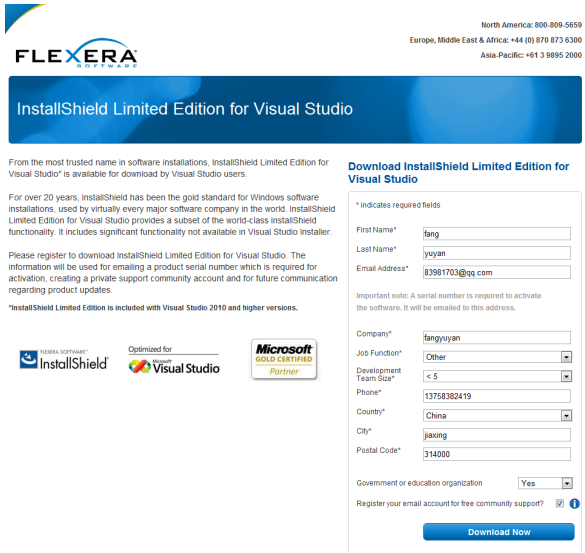

图 2-35　InstallShield 下载信息填写页面

（6）重新启动 VS2012。重复操作第（3）步，此时会发现在图 2-33 所示的添加新项目”对话框中多了“启动 InstallShield Limited Edition”项。选择“InstallShield Limited Edition Project”将跳出如图 2-37 所示的“InstallShield Product Activation”对话框，输入第（5）步中的激活码，完成产品激活。

（7）添加部署项目后，“project assistant”部署界面如图 2-38 所示，根据提示一步一步做好设置后运行部署项目即可。

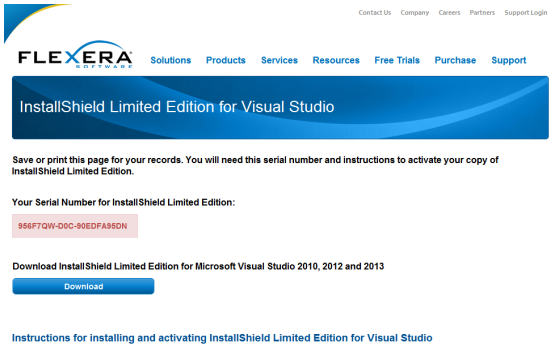

图 2-36 激活码显示及下载页面

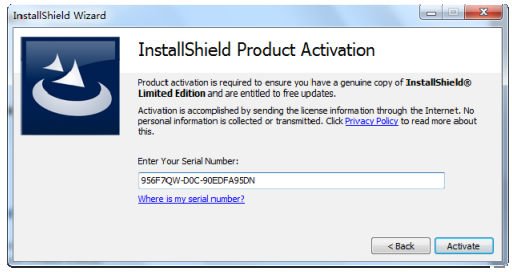

图 2-37 “InstallShield Product Activation”对话框

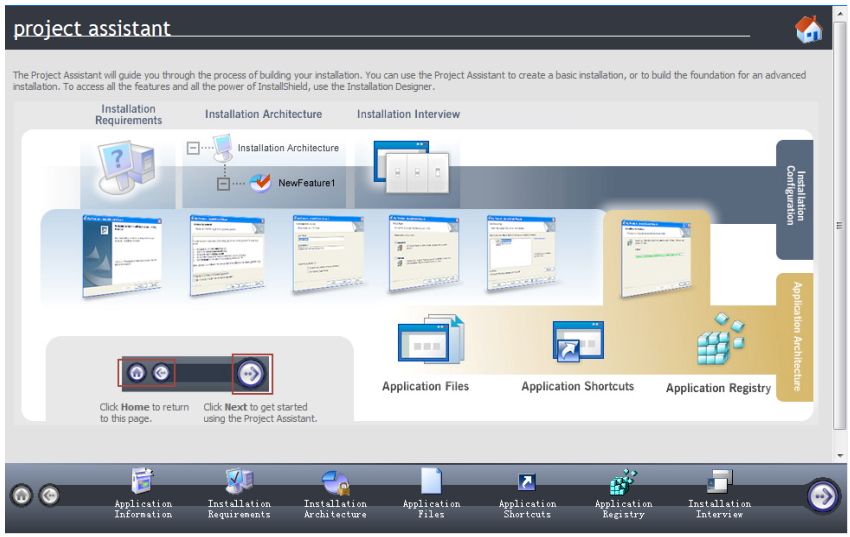

图 2-38 “project assistant”部署界面

生成解决方案后会在安装程序集下面生成一个名为 Setup 的文件夹，安装文件就保存在下面的目录中。Install 在生成安装文件后会有 Setup.exe 和.msi 两种安装文件，exe 文件是安装的引导文件，核心文件是 msi 文件，里面封存了程序的组件。在里面找到 Setup.exe 文件及 msi 文件即可进行安装。

最后还要强调一点，Install 打包工具中并没有继承中文环境，在它的底层语言库中没有中文语言，所以解决方案、程序集名称等涉及与主输出相关的文件最好不要以中文名命名，否则会出现如下的错误：“-7184: The FileName column of the File table includes characters that are not available on code page 1252”，该错误说明在代码段 1252 处没有发现和文件名相关的语言库。

网站打包成功后，在相应的文件夹中可以看到打包后的文件 addsetup.msi 和 setup.exe，双击 setup.exe 文件即可进行安装。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 2-21 所示。

表 2-21　演练完成情况评价表

任　务　号	2-13	任 务 名 称	网站发布与部署
任务子项	完成情况	主要问题	未完成原因
项目连接与测试			
部署环境的配置			
网站的打包			

三、知识点

1. 站点是按原样复制

Visual Web Developer 中的“复制网站”工具可以将文件从本地站点复制到远程站点，也可以实现从远程站点复制到本地站点；可以逐个复制选择的文件，也可以一次复制一个站点的所有文件，具体操作如下：

（1）在“网站”菜单上单击“复制网站”，打开如图 2-39 所示的“复制网站”窗口，在“源网站”列表中显示当前打开的网站中的文件。

（2）从“连接”列表中选择要作为远程站点进行连接的站点。如果要连接的站点不在列表中，请选择“连接”，然后使用“打开网站”对话框连接要复制文件的源站点或目标站点。

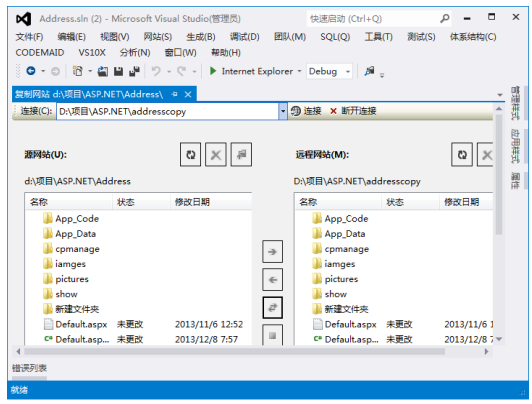

图 2-39　“复制网站”窗口

“复制网站”工具会在打开远程站点时检查两个站点上的文件并指示它们的状态（新建、未更改、已更改或已删除）。如果一个文件有两个版本，一个在源站点上，一个在远程站点上，则会有一个箭头从较新的版本指向较旧的版本。使用“复制网站”工具除了复制文件，还可以实现同步站点操作。

同步站点操作会检查本地站点和远程站点上的文件，并确保两个站点上的所有文件都是最新的。例如，如果远程站点上的某个文件比本地站点上同一文件的版本更新，同步文件功能会将远程站点上的文件复制到本地站点。

同步功能使得该工具非常适合用于多开发人员环境，在这种环境中，开发人员在本地计算机上保留网站的副本，各个开发人员可将其最新的更改复制到共享远程服务器，同时用其他开发人员提供的更改后的文件更新本地计算机。新加入的项目开发人员可以在自己的计算机上创建一个本地网站，然后与共享服务器上的站点进行同步，从而快速获取网站所有文件的副本。

网站原样复制常用的另外两种方式是把完成开发的网站复制到指定的目录，这种方式比较简单；另一种就是网站经过打包后，安装到指定的目录中。

采用站点按原样复制方式，比较简单，但也存在如下问题：

✧ 任何文件包含编译错误，直到有人（也许是用户）运行引发该错误的网页时才会发现；

✧ 随网站部署开发以外的人员也能看到的程序代码，增加了网站的风险。

2. 发布网站

使用 Microsoft Visual Web Developer Web 开发工具的“发布网站”实用工具来编译网站。

“发布网站”会对网站中的页和代码进行预编译，然后将编译器输出写入指定的文件夹，可以将输出复制到目标 Web 服务器，并从目标 Web 服务器中运行应用程序。

Visual Web Developer 可以发布网站，将编译网站输出复制到指定的位置，也可以采用直接把整个网站目录复制到目标 Web 服务器的方式。同简单地将网站复制到目标 Web 服务器相比，发布网站具有以下优点：

✧ 预编译过程能发现任何编译错误，并在配置文件中标识；

✧ 单独页的初始响应速度更快，因为页已经过编译，如果不先编译页就将其复制到网站，则将在第一次请求时编译页，并缓存其编译输出；

✧ 将 App_Code 文件夹中的页、源代码等预编译到可执行输出中，将可执行输出写入目标文件夹；

✧ 不会随网站部署任何程序代码，从而为文件提供了一项安全措施。

四、任务拓展

本节完成两个拓展实践任务，一个为课内实践任务，一个课外实践任务。

拓展任务卡 17

拓展任务号	2-17	任务名称	发布网站
计划用时	10 分钟	任务性质	课内
任务描述与目标			
Microsoft Visual Web Developer Web 开发工具的“发布网站”实用工具来编译网站，将网站通讯录发布到本机的虚拟目录上			
主要操作步骤提示			
1．打开本机 IIS，设置虚拟目录； 2．在 Visual Studio 2012 中打开通讯录网站，在“解决方案资源管理器”中右击项目名，在弹出的快捷菜单中选择“发布网站”选项，或在下拉菜单中执行“生成”→“发布网站”命令； 3．单击“发布网站”对话框的“目标位置”框中后面的…按钮，在弹出的“选择”对话框中选择“本地 IIS”，在 IIS 中选择刚设置的虚拟目录； 4．单击“确定”按钮，如果网站中不存在错误，则会显示“生成成功”，如果显示“生成失败”，那么重新打开网站进行调试			

拓展任务卡 18

拓展任务号	2-18	任务名称	复制网站
计划用时	30 分钟	任务性质	课外
任务描述与目标			
使用 Microsoft Visual Web Developer Web 开发工具的“复制网站”工具，进行网站部署			
主要操作步骤提示			
1．可以选择多个同学合作，设置局域网，并设置一个服务器共享目录； 2．在 Visual Studio 2012 中打开某个同学的通讯录网站； 3．使用 Visual Web Developer 中的“复制网站”工具，将网站复制到共享目录； 4．合作的同学打开 Visual Studio 2012，设置远程站点同步，并对网站文件进行更新； 5．练习共同开发			

拓展任务跟踪卡

任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

项目号			项目名	
项目完成方式			□小组协作 □个人独立	
项目完成情况（40%）	界面设计（2%）	自我评价		
		小组评价		
		个人评价		
	代码编写（20%）	自我评价		
		小组评价		
		个人评价		
项目完成情况（40%）	功能实现（10%）	自我评价		
		小组评价		
		个人评价		
	说明文档（5%）	自我评价		
		小组评价		
		个人评价		
	数据库设计(3%)	自我评价		
		小组评价		
		个人评价		
项目展示（30%）	语言表达(10%)	自我评价		
		小组评价		
		个人评价		
	团队合作(10%)	自我评价		
		小组评价		
		个人评价		
	展示形象(10%)	自我评价		
		小组评价		
		个人评价		
拓展与创新（30%）	创新能力（10%）	自我评价		
		小组评价		
		个人评价		
	设计潜力（10%）	自我评价		
		小组评价		
		个人评价		
主要存在问题				

课外思考题

1．如何通过代码来确定 Page_Load 事件是否因回发而触发运行？

2．请为下面用户输入字段确定使用什么类型的验证控件？

✧ 用户的年龄；

✧ 用户的电话号码；

✧ 用户的密码（需要输入两次）；

✧ 检查输入的数字是否为素数；

✧ 是否窗体里面所有的输入框都被正确填写；

✧ 日期的格式是否正确；

✧ 新员工的电子邮件地址是否符合公司的政策。

3．用户控件和组件的区别是什么？如何创建和使用用户控件？

4．怎么从宿主页访问一个用户控件 UI 元素的属性？

5．在同一个 ASP.NET 页中，可以使用两个相同名称的不同用户控件吗？为什么？

6．可以在两个不同的 Web 应用程序中使用同一个用户控件吗？

7．什么是.NET Framework 数据提供程序？.NET Framework 中包含哪些.NET 数据提供程序？

8．在 SQL Server 数据库连接时，要提供数据库连接安全模式，请问如何进行设置？各有什么特点？

9．DataReader 类是如何实例化的？什么时候使用 DataReader 类？

10．当不想返回行时，使用什么类型的语句？

11．在连接模式下 ADO.NET 中常用的对象有哪些？

12．如何处理.NET Framework 异常？

13．请说明在.NET 中常用的几种页面间传递参数的方法，并说出它们的优缺点。

14．ASP.NET 页面向服务器发送请求有几种方式？

项目三

信 息 验 证

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 了解 Web 网站登录、注册中可能存在的安全问题；
- ✧ 了解登录验证的必要性；
- ✧ 掌握数字验证的原理；
- ✧ 掌握图文验证的原理；
- ✧ 掌握 GDI+的基本知识；
- ✧ 掌握 SMTP 服务器的安装与配置；
- ✧ 掌握 System.Net.Mail 命名空间中与发送电子邮件相关的常用类的使用；
- ✧ 掌握第三方邮件收发组件 Jmail 组件的常用属性和常用方法。

技能目标

通过对本项目的学习，应该具备下面的能力：

- ✧ 能完成登录验证的代码实现；
- ✧ 能为网站选择合适的登录验证方式；
- ✧ 能利用 SMTP 服务器发送电子邮件；
- ✧ 能基于 Jmail 组件进行电子邮件的收发；
- ✧ 能利用所学的知识技能实现网站的用户注册邮件激活功能。

教学建议

本项目计划总学时为 10 学时。

- ✧ 情境介绍：0.5 学时；
- ✧ 任务 1：4.5 学时；
- ✧ 任务 2：5 学时。

教师在本项目开始授课时，可以通过网络展示网站中验证码的使用，并提出使用的原因，由此引申出本项目。

3.1 情境介绍

网络上一些黑客利用自动注册程序，大量地重复注册用户，并利用这些用户反复登录网站，以延缓服务器的响应速度，同时还可能对合法用户进行网络攻击，如发送大量垃圾邮件等。为了防止对这些自动注册程序的破坏，验证技术渐渐地发展起来。

在网站的信息验证方法中基本分验证码验证和邮件验证两种。

1. 验证码验证

验证码生成原理就是将一串随机产生的数字或符号，生成一幅图片，在图片里加上一些干扰像素，由用户肉眼识别其中的验证码信息，输入表单提交网站验证，验证成功后才能使用其中的功能。为了避免有些软件通过图形字符识别来获取验证码，一般会把验证码用各种颜色和不同大小，或者旋转一定角度来表示，有的还在里面掺杂各种杂点，这都是为了防止软件获取验证码。

验证码的实现流程是服务器端随机生成验证码字符串，保存在内存中，并写入图片，发送给浏览器端显示，浏览器端输入验证码图片上的字符，然后提交给服务器端，并确认提交的字符和服务器端保存的字符是否一致，一致就继续，否则返回提示。验证码的验证流程图如图 3-1 所示。

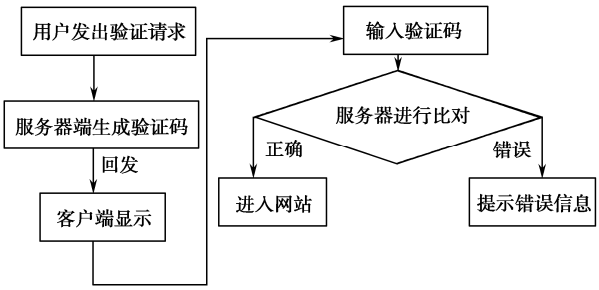

图 3-1 验证码的验证流程图

2. 邮件验证

当前的大多数网站均提供了用户注册功能，而流行的注册方式是注册新用户时，用户除了要填写用户名等相关信息外，还必须填写有效的电子邮箱地址，网站会自动生成一个随机的验证码，用户注册完成后，如果所填写的邮箱正确，会收到一封验证邮件，单击邮件中的链接，将会进行用户验证，如果通过验证，则用户就会拥有一个可以登录的账号。

如图 3-2 所示给出了基于电子邮件激活功能的用户注册和登录功能流程图。

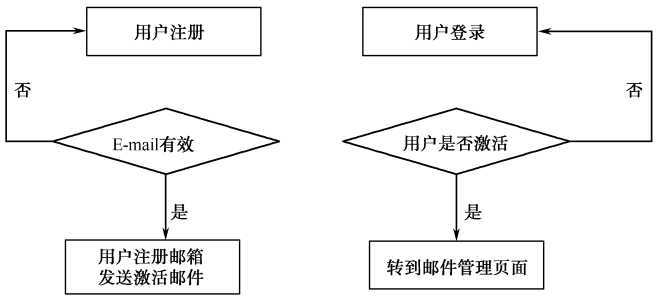

图 3-2 用户注册和登录功能流程图

3.2 任务 1 ASP.NET 图文处理

任务描述

随着验证技术的发展，目前各站点都使用了验证码，随机数字验证方式是最简单也是最原始的方式，本任务通过验证学习如何在 ASP.NET 中运用随机数生成、字符串处理和图形处理。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）随机数的生成；

（2）数字验证功能的实现。

（3）生成随机图文；

（4）实现图文验证功能。

3.2.1 .NET 伪随机数生成器

随机数的生成是数字验证的基础，.NET 伪随机数生成器是一种能够产生满足某些随机性统计要求的数字序列的类。

一、实战演练

（1）打开“Address”网站，在“解决方案资源管理器”中右击“App_Code”目录，添加一个类文件 PDValidate.cs。

（2）双击打开类文件，在类文件中添加获取随机数的方法，如代码清单 3-1 所示。

代码清单 3-1 获取随机数的方法代码

```
public class PDValidate
{
    int n;
    public PDValidate(int x)
    {       n = x;      }
    public string Getcode3()
    {
        string vnum = "";
        int temp = -1;
        Random rand = new Random();
        for (int i = 1; i <= n; i++)
        {
            if (temp != -1)
              rand = new Random(i * temp * unchecked((int)DateTime.Now.Ticks));
            int t = rand.Next(10);
            temp = t;
            vnum += t.ToString();
        }
    }
}
```

（3）按【Shift+F6】组合键生成网站，以检查有无语法错误。

（4）打开网站中的 login.ascx 文件，在用户名密码中拖入一行，拖入一个名为“ValidateCode”的 TextBox 控件，再分别拖入 RequiredFieldValidator 和 CustomValidator，把它们的 ControlToValidate 属性设为“ValidateCode”，Display 属性设为“None”，再分别设置 ErrorMessage 的值为“验证码必填”和“验证码错误”。

拖入一个 ValidationSummary 控件用于显示验证错误消息，再添加一个 Label 控件，为其添加一个用于设置背景的 CSS 样式。

（5）进入“源”视图，添加 Page_Load 事件，如代码清单 3-2 所示。

代码清单 3-2　Page_Load 事件代码

```
protected void Page_Load(object sender, EventArgs e)
    {
        if(!Page.IsPostBack)
        {  PDValidate pdv=new PDValidate(6);
        Label1.Text = pdv.GetVcode();
        }
 }
```

（6）添加自定义验证控件的 CustomValidator1_ServerValidate 事件，如代码清单 3-3 所示。

代码清单 3-3　CustomValidator1_ServerValidate 事件代码

```
    {
        if (ValidateCode.Text.Trim() == Label1. Text.
Trim())
        {
            args.IsValid = true;
        }
        else args.IsValid = false;
    }
```

（7）运行程序，输入正确与错误的验证码进行测试，添加了验证码的登录用户设计界面如图 3-3 所示。

图 3-3　验证界面

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-1 所示。

表 3-1　演练完成情况评价表

任　务　号	3-1	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
创建类文件 PDValidate			
添加获取随机数的方法			

三、知识点

在图文验证中最关键的是随机数的生成，生成随机数使用的是.NET 所提供的 Random 类。Random 类是伪随机数生成器，能够产生满足某些随机性统计要求的数字序列。伪随机数是以相同的概率从一组有限的数字中选取的，所选数字并不具有完全的随机性，因为它们是用一种确定的数学算法选择的，但是从实用的角度而言，其随机程度已足够了。

随机数的生成是从种子值开始的，如果反复使用同一个种子，就会生成相同的数字系列。产生不同序列的一种方法是使种子值与时间相关，从而对于 Random 类中的每个新实例，都会产生不同的系列。默认情况下，Random 类的无参数构造函数使用系统时钟生成其种子值，而

参数化构造函数可根据当前时间的刻度数采用 Int32 值。

在创建 Random 对象时采用不同的构造函数产生不同的随机数生成器。

✧ Random()构造函数

生成的数字分布均匀，这样每个数字返回的可能性均相等。默认种子值是从系统时钟派生而来的，具有有限的分辨率。因此，通过调用默认构造函数而频繁创建的不同 Random 对象将具有相同的默认种子值，因而会产生几组相同的随机数。为不同的 Random 对象提供相同的种子值会导致每个实例产生相同的随机数序列。

✧ Random(Int32)构造函数

指定的种子值初始化 Random 类的新实例。如果应用程序需要不同的随机数序列，则可以用不同的种子值重复调用此构造函数。一种产生唯一种子值的方法是使它与时间相关。

✧ Random.Next()方法

Next()方法有如下三个重载，每个重载生成随机数的方式不同。

Next()生成一个值，范围是 0 与 Int32.MaxValue 之间的随机数。

Next(maxValue)返回一个小于 maxValue 所指定最大值的非负随机数。

Next(minValue, maxValue)返回一个大于等于 minValue 且小于 maxValue 的 32 位带符号整数。

3.2.2 .NET 基本字符串操作

System.String 和 System.Text.StringBuilder 类中的多个方法可以动态构造要在用户界面中显示的自定义字符串。这些方法也执行许多基本字符串操作，如从字节数组创建新字符串，比较字符串的值和修改现有的字符串。

一、实战演练

（1）打开“Address”网站。

（2）在“解决方案资源管理器”中右击“App_Code”目录，添加一个类文件添加图文验证类 ImageDValidate。

（3）在类中添加 GetCode()方法，如代码清单 3-4 所示。

代码清单 3-4 添加图文验证类及 GetCode()方法代码

```
public class ImageDValidate
{  int codeLenth;
   public ImageDValidate(int x)
   {       codeLenth = x;         }
   public string GetCode()
   {
    string a = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789";
    StringBuilder sb = new StringBuilder();
    //随机生成字符串
    for (int i = 0; i < codeLenth; i++)
    {
    sb.Append(a[new Random(Guid.NewGuid().GetHashCode()).Next(0, a.Length - 1)]);
     }
        return sb.ToString();
    }
}
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-2 所示。

表 3-2 演练完成情况评价表

任 务 号	3-2	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
生成随机数字			

三、知识点

在网站中对字符串的进行处理是非常常见的操作处理。

1. 字符串基本操作

.NET Framework 在 System.String 类中提供了许多字符串操作方法，下面的方法是在应用程序开发中常用的。

✧ Format()方法

此方法使用 .NET Framework 的复合格式设置功能将对象的值转换为其文本表示形式，并将该表示形式嵌入字符串中。

该方法包含多个函数重载，但可以归纳为：

```
Format(string format, Object[] args)
```

前面的一个参数的作用是指定复合格式字符串，后面需要进行格式化的对象，例如：

```
//显示为 2013/12/17,2013 年 12 月 17 日,2013/12/17 16:56,2013/12/17 16:56:01
String.Format("{0:d},{1:D},{2:g},{3:G}", DateTime.Now, DateTime.Now,DateTime.
Now, DateTime.Now);
```

此方法对一些信息以特殊的方法显示的设置非常方便。

✧ Insert ()方法

此方法在指定索引位置插入一个指定的字符串。格式为：

```
string Insert(int startIndex,string value)
```

✧Trim ()方法

该方法包含 2 个重载，从当前字符串对象中移除一组指定字符的所有前导匹配项和尾部匹配项，如果没有指定移除的字符串，则移除空串。格式为：

```
string Trim(char[] trimChars)
```

✧ Remove ()方法

该方法包含二个重载，从此实例中的指定位置开始删除指定数目的字符，如果不指定删除的字符数，则删除到字符串最后。

```
string Remove(int startIndex, int count)
```

✧ Compare ()方法

比较两个指定的 String 对象，该方法包含多个重载，最常见的方法格式是：

```
int Compare(string strA, string strB, bool ignoreCase
```

✧ ToUpper ()方法和 ToLower ()方法

这二个方法是把指定的字符串转换为大写或小写。

2. 高效操作字符串

字符串处理是在软件开发中经常会面对的问题。在.NET 中，字符串是 Char 数据类型的字符数组，在赋值后，将无法改变其值。如果要改变字符串的值，则必须创建一个新的字符串，这就需要为该新对象分配新的空间，使变量指针指向新的字符串。在需要对字符串执行重复修改的情况下，与创建新的 String 对象相关的系统开销可能会非常昂贵。

如果要修改字符串而不创建新的对象，则可以使用 System.Text.StringBuilder 类。例如，当在一个循环中将许多字符串连接在一起时，使用 StringBuilder 类可以提升性能，可以在不创建新对象的情况下，从字符串中追加、替换和删除字符。StringBuilder 类属于 System.Text 命名空间。

创建 StringBuilder 类的实例化方法如下：

```
StringBuilder MyStringBuilder = new StringBuilder("Hello World!");
```

StringBuilder 类常用的属性和方法如下。

✧ Chars 属性：获取或设置此实例中指定字符位置处的字符，使用方式如 sb. Chars[1]；

✧ Length 属性：获取或设置当前实例长度；

✧ Append()方法：可用来将文本或对象的字符串表示形式添加到由当前 StringBuilder 对象表示的字符串的结尾处，如 sb.Append(" What a beautiful day.")；

✧ Insert()方法：将字符串或对象添加到当前 StringBuilder 对象中的指定位置，如 sb.Insert (6,"Beautiful ")；

✧ Remove()方法：从当前 StringBuilder 对象中移除指定数量的字符，移除过程从指定的 0 索引处开始，如 Sb.Remove(5,7)；

✧ Replace()方法：可以用另一个指定的字符来替换 StringBuilder 对象内的字符，如 Sb.Replace('!', '?')；

✧ AppendFormat()方法：将经过指定的格式化的标准格式字符串文本添加到 StringBuilder 对象的末尾，如 sb.AppendFormat("{0:C} ", 25)。

尽管 StringBuilder 类在操作字符串时能提高性能，但创建一个 StringBuilder 实例比创建一个 String 对象更消耗内存资源。因此，当只需要少量的字符串追加时，最好使用 String 类，但当需要多次追加字符串时，如在循环中使用 StringBuilder 类比使用 String 类更能提高程序的性能。

3. 复合格式设置字符串

复合格式设置功能使用对象列表和复合格式字符串作为输入格式的设置。复合格式字符串由固定文本和索引占位符混和组成，其中索引占位符称为格式项，对应于列表中的对象。格式设置操作产生的结果字符串由原始固定文本和列表中对象的字符串表示形式混和组成。

复合格式设置功能受 Format、AppendFormat 以及 WriteLine 和 TextWriter.WriteLine 某些重载方法支持。String.Format 方法可产生设置了格式的结果字符串，AppendFormat 方法将设置了格式的结果字符串追加到 StringBuilder 对象，Console.WriteLine 方法将设置了格式的结果字符串显示到控制台，TextWriter.WriteLine 方法将设置了格式的结果字符串写入流或文件。

复合格式字符串由零个或多个固定文本段与一个或多个格式项混和组成。固定文本是所选择的任何字符串，并且每个格式项对应于列表中的一个对象或装箱的结构。复合格式设置功能返回新的结果字符串，其中每个格式项都被列表中相应对象的字符串表示形式取代。

每个格式项都采用下面的形式并包含以下项目：

{ 索引[,对齐][:格式字符串]}

必须使用成对的大括号（“{”和“}”）。

复合格式字符串中针对不同的数据类型，格式设置的含义不同，表 3-3 列出了常用格式字符串的含义及用法。

表 3-3 常用格式字符串含义说明表

数据类型	格式说明符	名称	实例及说明
数字格式字符串	C 或 c	货币	数字转换为表示货币金额的字符串。如 String.Format("{0:C}", 12345.6789);//显示¥12,345.68

	#	数字占位符	设置数字点位，如
			String.Format("{0:#}", 12345.6789); //显示 123456
			String.Format("{0:[##-##-##]}", 12345.6789); //显示[1-23-46]

	,	千位分隔符	与其他字符一起设置千分位，如 String.Format("{0:#,#.00}", 12345.6789); //显示 12,345.68
	.	小数点	设置小数点分隔符的位置，如：String.Format("{0:0.00}", 12345.6789); //显示 12345.68

日期和时间格式字符串	D /d	长短日期	String.Format("{0:d}", DateTime.Now); //显示 2013/12/19
			String.Format("{0:D}", DateTime.Now); //显示 2013 年 12 月 19 日

	F/f	完整长短日期	//显示 2013 年 12 月 19 日 11:06:38
			String.Format("{0:F}", DateTime.Now);
			//显示 2013 年 12 月 19 日 11:07
			String.Format("{0:f}", DateTime.Now);

	G/g	常规长短日期	String.Format("{0:G}", DateTime.Now); //显示 2013/12/19 11:08:32
			String.Format("{0:g}", DateTime.Now); //显示 2013/12/19 11:09

	多个 d		String.Format("{0:dd}", DateTime.Now); //显示 19
			String.Format("{0:ddd}", DateTime.Now); //显示周四
			String.Format("{0:dddd}", DateTime.Now); //显示星期四

3.2.3 .NET 图形处理

在 ASP.NET 信息处理中，经常会涉及一些图形的处理问题，比如给上传的图片进行水印处理，信息上传后转换为图片的方式输出等问题。在图文验证中最关键的也是图形的生成。通过图形的验证码生成学习.NET 图形图像处理类。

一、实战演练

（1）打开“Address”网站。

（2）打开图文验证类 ImageDValidate，添加 GetImage()方法，生成图文，如代码清单 3-5 所示。

代码清单 3-5 GetImage()方法代码

```
    public Bitmap GetImage(string str)
        {
            Font  codefont ;
           //存放不同的字体类型，以便生成不同形状的验证码
            string[] strfont = new string[] { "Trebuchet MS", "Bauhaus 93", "Bernard
MT Condensed", "Blackoak Std", "Arial", "Brush Script MT" };
            int codewidth = (int)(str.Length * 20);
           //创建一个高为 50，宽与字符个数相关长度的图像对象
            Bitmap images = new Bitmap(codewidth, 50);
            Graphics grap1 = Graphics.FromImage(images);
            Brush bs = new SolidBrush(Color.Black);
            grap1.Clear(Color.Pink);
            Random r = new Random();
            //生成每个字符不同高度、字形的验证码图片
           int codesize = 10, fontindex= 0, loca = 5;
            for (int i = 0; i < str.Length; i++)
            {
                 fontindex = r.Next(0, 5);
                if (i % 2 == 0)
                {
                   codesize = r.Next(12, 20);
                 codefont = new Font(strfont[fontindex], codesize, FontStyle.Italic);
                }
                else
                {
                   codesize = r.Next(14, 22);
                  codefont  =  new  Font(strfont[fontindex],  codesize,  FontStyle.
Regular);
                }
              grap1.DrawString(str[i].ToString(), codefont, bs, loca, r.Next(1, 20));
                loca = (int)(loca + codesize * 1.1);
            }
            Pen bp = new Pen(Color.Bisque, 0);
            Random r1 = new Random();
            //生成图片的雪花点作为图片噪声
            for (int i = 0; i < 200; i++)
            {  int x1 = r.Next(0, images.Width);
              int y1 = r.Next(0, images.Width );
               if (i % 2 == 0)
               {
                  grap1.DrawLine(bp, x1, y1, x1 + 1, y1);
               }
               else
               {
                  grap1.DrawLine(bp, x1, y1, x1, y1 + 1);
               }
            }
           return images;
        }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-4 所示。

表 3-4 演练完成情况评价表

任 务 号	3-3	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
生成图文验证图像			

三、知识点

Web 应用程序的开发经常会涉及图形图像的处理，在.NET 中使用 GDI+类库来创建图形，GDI+将很多处理图形图像的类和接口封装在 System.Drawing 命名空间中。Graphics 类是 GDI+的核心功能，它是实际绘制直线、曲线、图形、图像和文本的类。在实际应用中 Graphics 类创建一个绘制表面，使用 Brush、Pen、Image、Icon、Bitmap 等类在绘制表面上绘制图形。其中，Pen 类可用于绘制直线和曲线；Brush 类可用于为形状填充颜色；Color 类可用于指定要用的颜色；Size 类可用于控制形状的大小；Image 类和 Bitmap 类用于显示和操作绘制图像；Icon 类可用于管理应用程序中的图标。如图 3-4 所示显示了绘制过程。

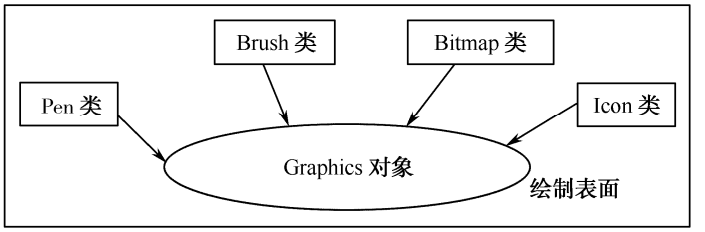

图 3-4 绘制过程

1. Graphics 类

Graphics 类提供将对象绘制到显示设备的方法。Graphics 类与特定的设备上下文关联。创建 Graphics 类常用以下两种方法：

✧ 调用 Control.CreateGraphics()方法来创建 Graphics 对象，创建方式如：

```
Graphics g=this. CreateGraphics();
```

✧ 使用 Graphics.FromImage()方法从图像创建 Graphics 对象，创建方式如：

```
Image imageFile = Image.FromFile("Imag.jpg");
Graphics newGraphics = Graphics.FromImage(imageFile);
```

Graphics 类的方法有很多，最常用的方法主要是进行各类图形的绘制，如 DrawLine()方法用于绘制一条线。具体的 Graphics 类的方法可以参考 MSDN 中.NET Framework 类库的 Graphics 类。

2. Pen 类

定义用于绘制直线和曲线的对象。在任何图形中，直线和曲线都是最重要的组成部分。通过 Pen 类，可以使用直线和曲线来绘制独立图形和其他复合图形的轮廓，直线可以是不同宽度和不同样式。

最常用的 Pen 类实例化格式如：

```
Pen myPen = new Pen(Color.Blue, 2)
```

其常用属性如下。

✧ Color 属性：获取或设置 Pen 实例的颜色；

✧ Width 属性：获取或设置此 Pen 实例的宽度；

✧ Brush 属性：获取或设置 Brush，用于确定此 Pen 的属性；

✧ EndCap 属性：获取或设置要在通过此 Pen 绘制的直线终点使用的线帽样式；

✧ StartCap 属性：获取或设置在通过此 Pen 绘制的直线起点使用的线帽样式。

3. Brush 类

定义用于填充图形形状（如矩形、椭圆、饼形、多边形和封闭路径）内部的对象。Brush 类是抽象基类，不能进行实例化。若要创建一个画笔对象，应该使用从 Brush 派生出的类，如 SolidBrush、TextureBrush 和 LinearGradientBrush 类。

✧ SolidBrush 类定义单色画笔，实例化格式如：

```
SolidBrush sBrush = new SolidBrush(Color.Red);
```

✧ TextureBrush 类使用图像来填充形状的内部，实例化格式如：

```
Bitmap image1 = (Bitmap) Image.FromFile(@"back.gif", true);
TextureBrush tBrush = new TextureBrush(image1);
```

✧ LinearGradientBrush 类使用双色渐变和自定义多色渐变填充形状的内部，实例化格式如：

```
LinearGradientBrush lBrush=new LinearGradientBrush(0,0, Color.Red, Color.
Blue);
```

4. Bitmap 类

Bitmap 类用于处理由像素数据定义的图像对象。位图由图形图像及其属性的像素数据组成，可使用许多标准格式将位图保存到文件。GDI+支持的文件格式有 BMP、GIF、EXIG、JPG、PNG 和 TIFF。可以使用 Bitmap 构造函数从文件、流和其他源创建图像，然后使用 Save 方法将这些图像保存到流或文件系统中，常见格式如：

```
Bitmap image1=new Bitmap (@"back.gif")
```

其常见的属性与方法如下。

✧ RawFormat 属性：获取此 Image 的文件格式，其值为 ImageFormat；

✧ Size 属性：获取此图像以像素为单位的宽度和高度；

✧ Width 属性：获取此 Image 的宽度；

✧ Height 属性：获取此 Image 的高度；

✧ SetPixel()方法：设置指定像素的颜色；

✧ Save()方法：将指定图片以某种方式保存到指定的文件或流；

✧ GetPixel()方法：获取指定像素的颜色。

上面列举的这几个类是 GDI+关键类，GDI+还定义了许多辅助类，常用的如 Color 类用于设置颜色；Font 类可以绘制表面写入文本；Rectangle 类绘制图形形状；Point 类和 Size 类指定大小。相关的类还很多，不一一列举，有兴趣的读者可以参考 MSDN 的相关内容及相关书籍。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 1

拓展任务号	3-1	任务名称	扩展图文验证用的声噪
计划用时	30 分钟	任务性质	课内

任务描述与目标

能根据自己的实际需要设计出不同的噪声，并通过中英文验证码生成器不仅灵活掌握验证码的生成原理，同时进一步掌握文件与 GDI+在项目开发中的应用

主要操作步骤提示

改写 GetImageCode()，要求把原来的雪花点噪声改为一条曲线，参考代码如下：

```
public Bitmap GetImage(string str)
{
 //num1 获取文字长度计算出图片宽度
  int num1 = Convert.ToInt32((double)(str.Length * 17));
  int num2 = 30; //num2 图片高度
Bitmap bitmap1 = new Bitmap(num1, num2);  //定义图片
Graphics graphics1 = Graphics.FromImage(bitmap1);  //定义画板
graphics1.Clear(Color.White);  //清空画板
 //文字颜色集
  Color[] colorArray1 = new Color[] { Color.Black, Color.Red, Color.DarkBlue, Color.Green,
Color.Brown, Color.DarkCyan, Color.Purple };
  //线条颜色集
 Color[] colorArray2 = new Color[] { Color.LightBlue, Color.LightCoral, Color.LightCyan, Color.
LightGoldenrodYellow, Color.LightGray, Color.LightGreen, Color.LightPink, Color.LightSalmon,
Color.LightSeaGreen, Color.LightSkyBlue, Color.LightYellow };
   Random random1 = new Random();  //定义随机变量
Pen pen1 = new Pen(Color.LightBlue, 0f); //定义画笔
    int num3 = 20; //定义线条数
   for (int num4 = 0; num4 < num3; num4++)
    {
           pen1 = new Pen(colorArray2[random1.Next(11)], 0f); //定义画笔颜色
           //定义四点
           Point point1 = new Point(random1.Next(num1), random1.Next(num2));
           Point point2 = new Point(random1.Next(num1), random1.Next(num2));
           Point point3 = new Point(random1.Next(num1), random1.Next(num2));
           Point point4 = new Point(random1.Next(num1), random1.Next(num2));
           graphics1.DrawBezier(pen1, point1, point2, point3, point4); //画出曲线
    }
       //存放字符
       string text1 = string.Empty;
       //字符位置每次间隔 15
       int num5 = 2;
       for (int num6 = 0; num6 < Str.Length; num6++)
     {
text1 = Str.Substring(num6, 1);    //获取字符
//画出字符字体画笔 XY
     graphics1.DrawString(text1,  new  Font("Arial",  16f,  FontStyle.Bold),  new  SolidBrush
(colorArray1[random1.Next(6)]), (float)num5, 2f);
           num5 += 15;  }
return bitmap1;
}
```

3.2.4 ASP.NET 流信息

在 ASP.NET 信息处理中，经常会涉及一些文件信息的处理问题。文件是一些具有永久存储及特定顺序的字节组成的一个有序的、具有名称的集合。与文件操作相关的有目录路径、磁盘存储、文件和目录名等方面。相反，流提供一种向后备存储写入字节和从后备存储读取字节的方式，后备存储可以为多种存储媒介之一，除文件流之外也存在多种流，例如，网络流、内存流和磁带流等。

一、实战演练

（1）打开“Address”网站。

（2）打开图文验证类 ImageDValidate，添加 GetImageCode()方法，生成图文，如代码清单 3-6 所示。

代码清单 3-6 GetImage()方法代码

```
public MemoryStream GetImageCode( string str)
    {
        Bitmap images = GetImage(str);
        MemoryStream ms = new MemoryStream();
        images.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg);
        return ms;
    }
```

（3）右击“Address”网站，添加 showimages.aspx 文件，文件中不添加任何控件，在 CS 文件中添加 Page_Load 事件，该文件主要是实现验证码以图形的方式显示。代码清单如 3-7 所示。

代码清单 3-7 showimages.aspx 文件的 Page_Load 事件

```
protected void Page_Load(object sender, EventArgs e)
    {
        ImageDValidate idv=new ImageDValidate(6);
        string valicode=idv.GetCode();
        //用于验证时比对
        Session["valicode"] = valicode;
        MemoryStream ms = idv.GetImageCode(valicode);
        Response.ClearContent();
        //向页面输出图片的字节流，生成图片
        Response.ContentType = "image/Jpeg";
        Response.BinaryWrite(ms.ToArray());
    }
```

（4）打开 regist.aspx 文件，修改文件中的 HTML，在登录按钮上方添加一行，标记代码清单如下如 3-8：

代码清单 3-8 添加验证码行

```
<tr>
<td class="tdleft">验证码
</td>
<td class="tdright">
    <asp:TextBox ID="ValidateCode" runat="server" Width="60px"></asp:TextBox>
    <img src="showimages.aspx" id="ValidateCodeImage" style="height: 30px"
onclick="refresh();" alt="看不清，点击换一张" />
    <asp:RequiredFieldValidator ID="RequiredFieldValidator5" runat="server"
CssClass="yz"
```

```
            ControlToValidate="ValidateCode" Display="None" ErrorMessage="
    验证码必填"></asp:RequiredFieldValidator>
       <asp:CustomValidator  ID="CustomValidator2"  runat="server"  Display="
None"
     ControlToValidate="ValidateCode" ErrorMessage="验证码错误" CssClass="yz"

OnServerValidate="CustomValidator2_ServerValidate" ></asp:CustomValidator>
      </td>
     </tr>
```

（5）在 regist.asp 文件上方添加 refresh()脚本事件，代码清单如 3-9 所示。

代码清单 3-9　refresh()脚本事件

```
    <script language="javascript" type="text/javascript">
     function refresh() {
       var url = "showimages.aspx?id=";
    var r = Math.random() * 1000;
       url = url + r;
     document.getElementById("ValidateCodeImage").src = url;
    document.all("UserCode").value = "";
       document.all("UserCode").focus();
       return false;
       }
    </script>
```

（6）添加 CustomValidator2_ServerValidate 事件中的代码，事件代码如下。

```
    if (ValidateCode.Text.Trim().ToUpper() == Session["valicode"].ToString().
ToUpper())
           {
              args.IsValid = true;
           }
           else args.IsValid = false;
```

（7）运行程序，输入正确与错误的验证码进行测试，添加了图文验证的部分注册页面如图 3-5 所示。

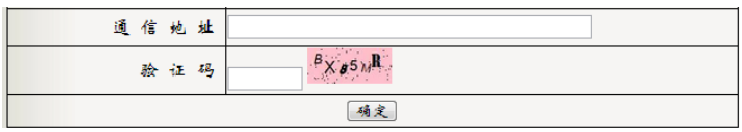

图 3-5　添加验证的注册页面

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-5 所示。

表 3-5　演练完成情况评价表

任　务　号	3-4	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
在注册页面添加图文验证			

三、知识点

操作系统在磁盘中存储文件，同一个文件的数据可能并不是连续分布在磁盘上的。如何读

取分散的文件数据呢？答案是流。流是一个数据字节序列，是两个端点之间的管道，它被用于访问文件的内容。在.NET 中提供 Stream 类来读/写文件或内存中的文件数据流。

Stream 类是一个抽象类，它提供了.NET 中所有流类的基本功能。FileStream、MemoryStream 和 BufferedStream 这三个类从 Stream 类继承而来，都属于 System.IO 命名空间，在这里主要介绍 FileStream 和 MemoryStream 两个类。

1. 使用 FileStream 类管理文件数据

FileStream 类主要用于文件系统上的文件进行读取、写入、打开和关闭操作。如代码清单 3-10 所示演示了 FileStream 类的基本用法。

代码清单 3-10　FileStream 类的基本用法

```
string overview = "Most commercial applications, such as...;
FileStream conduit = new FileStream(("Overview.txt ", FileMode.Create);
byte[] encodedOverview = new UTF8Encoding(true).GetBytes(overview);
conduit.Write(encodedOverview, 0, encodedOverview.Length);
conduit.Close();
```

其常用的属性和方法如下。

✧ Length 属性：获取用字节表示的流长度；

✧ Position 属性：获取或设置此流的当前位置；

✧ Read()方法：从流中读取字节块并将该数据写入给定缓冲区中，返回读入缓冲区中的总字节数。如果当前的字节数没有所请求得那么多，则总字节数可能小于所请求的字节数；或者如果已到达流的末尾，则为 0；

✧ Seek()方法：将该流的当前位置设置为给定值，返回流中的新位置；

✧ Write()方法：使用从缓冲区读取的数据将字节块写入该流；

✧ Close()方法：关闭当前流并释放与之关联的所有资源。

2. 使用 MemoryStream 类管理内存数据

MemoryStream 类用于在内存中写入、读出数据，存储在内存中的数据不会被永久保存。使用 MemoryStream 类从内存中读取、写入数据的性能优于使用 FileStream 类从文件中读取、写入数据。如果是处理数据时需要临时使用数据，应该将其存储在内存而不是文件中。MemoryStream 类的常用属性和方法与 FileStream 类类似，这里不重复介绍。如代码清单 3-11 演示了 MemoryStream 类的基本用法。

代码清单 3-11　MemoryStream 类的基本用法

```
    byte[] overview = new UTF8Encoding(true).GetBytes("Most commercial applications,
such as...");
    MemoryStream conduit = new MemoryStream(overview.Length);
    conduit.Write(overview, 0, overview.Length);
    Console.WriteLine(conduit.Position.ToString());
    conduit.Flush();
    conduit.Seek(0, SeekOrigin.Begin);
    byte[] overviewRead = new byte[conduit.Length];
    conduit.Read(overviewRead, 0, ((int)conduit.Length));
    Console.WriteLine(new UTF8Encoding().GetChars(overviewRead));
    conduit.Close();
```

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 2

拓展任务号	3-2	任务名称	为网络通讯录登录添验证码验证
计划用时	30 分钟	任务性质	课内

任务描述与目标

通过与已开发网站的实际结合，灵活掌握图文验证功能

主要操作步骤提示

1．建立一个文本文件，把中英文文本存放在其中；

2．改写 GetCode()方法，随机生成的验证码改为从文本文件读取；

```
public string GetCode(string txtFilePath)
    {
        string txtFullPath = HttpContext.Current.Server.MapPath("~/") + txtFilePath;
        FileInfo fi = new FileInfo(txtFullPath);
        string s = "";
        StreamReader sr = fi.OpenText();
        s = sr.ReadLine();
        StringBuilder sb = new StringBuilder();
        //随机生成字符串
        for (int i = 0; i < n; i++)
        {
            sb.Append(s[new Random(Guid.NewGuid().GetHashCode()).Next(0, s.Length - 1)]);
        }
        return sb.ToString();
    }
```

3．打开网络通讯录；

4．在“解决方案资源管理器”中为通讯录的登录用户控件添加图文验证功能

3.3 任务 2 ASP.NET 邮件处理

任务描述

当前的大多数网站均提供了用户注册功能，而流行的注册方式是注册新用户时，用户除了要填写用户名等相关信息外，还必须填写有效的电子邮箱地址，网站会自动生成一个随机的验证码，用户注册完成后，如果所填写的邮箱正确，会收到一封验证邮件，单击邮件中的链接，将会进行用户验证，如果通过验证，则用户就会拥有一个可以登录的账号。

通过本任务将学习如何使用 ASP.NET 提供的邮件收发收邮件，同时学习邮件第三方组件的使用。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）学习使用.NET 平台的邮件发送类实现发送电子邮件；

（2）学会如何激活注册用户。

3.3.1 使用 ASP.NET 类实现电子邮件的发送

ASP.NET 提供了实现在 Web 应用程序中的邮件收发功能，.NET 平台提供了邮件发送类，这些类集中在 System.Net.Mail 命名空间中。

一、实战演练

（1）打开“Address”网站的数据库，在用户信息表的 userinfo 中加入两个新的字段，字段名分别是 isactive 和 code，类型分别为 bit 和 char，用于表示该账号是否被激活和验证码的存放，其中 isactive 的默认值为 0。改进后的注册用户信息表（userinfo）如图 3-6 所示。

列名	数据类型	允许空
userid	int	☐
username	varchar(20)	☐
userpassw	varchar(100)	☐
realname	varchar(50)	☑
sex	nchar(10)	☑
email	varchar(50)	☐
phone	char(20)	☑
headpicture	varchar(100)	☑
datetime	datetime	☑
hobby	nvarchar(50)	☑
birthday	datetime	☑
province	nchar(10)	☑
isactive	bit	☑
code	char(200)	☑

图 3-6　注册用户信息表（userinfo）

（2）打开网络通讯录网站 address，设计一个网络通讯录的使用公约的文本文件存放在“show”文件夹下。

（3）打开 App_Code 文件夹下的 StringControl.cs 文件，添加 getcode()方法，方法代码如代码清单 3-12。

代码清单 3-12　获取用户注册激活码

```
    string getcode(string username,string userpass, string usermail)
    {
    //把用户名密码及用户注册邮件设置为激活凭证
       string input = String.Format("{0}|{1}|{2}", username, userpass, usermail);
     string code = common.MD5(input);
       return code;
    }
```

（4）在 App_Code 文件夹下添加 MailControl.cs 文件，添加 sendMail ()方法，方法代码如代码清单 3-13。

代码清单 3-13　发送激活邮件

```
    void string sendMail(string username, string usermail,string code)
    {
    string txtFullPath = HttpContext.Current.Server.MapPath("~/") + "show/网络通讯录系统公约.txt";
           // 创建一个附件对象
    Attachment objMailAttachment = new Attachment(txtFullPath);
    // 创建邮件消息
    MailMessage objMailMessage = new MailMessage();
    objMailMessage.From = new MailAddress("mi_fyyple@163.com");
    //源邮件地址
    objMailMessage.To.Add(usermail);          //注册人的邮件地址
    objMailMessage.Subject = "账户激活通知!";  //发送邮件的标题
    //将附件附加到邮件消息对象中
    objMailMessage.Attachments.Add(objMailAttachment);
    //激活文件的地址目前为本机地址，发布前改为服务器地址
          string result = string.Format(@"http://localhost/address/ActivateUser.aspx?
    UserName={0}&Code={1}", username, code);
           string body = String.Format(@"亲爱的用户 {0}：您好！感谢您注册网络通讯录，您只需要单击下面的链接，激活您的账户，便可以享受本网站提供的各项服务。
http://localhost/address/ActivateUser.aspx?UserName={1}&amp;Code={2}
(如果无法单击该 URL 链接地址，请将它复制并粘贴到浏览器的地址输入框中，然后按【Enter】键即可
", username, username, code);
           objMailMessage.Body = body;       //发送邮件的内容
           objMailMessage.IsBodyHtml = true;
           SmtpClient SmtpMail = new SmtpClient();
           //设置发件箱的 SMTP 服务器
           SmtpMail.Host = "smtp.163.com";
```

```
            //SMTP 使用的端口
            SmtpMail.Port = 25;
            //使用邮箱登录名和密码的验证
            SmtpMail.Credentials = new NetworkCredential("usermailloginname",
"usermailpass");
            SmtpMail.Send(objMailMessage);
    }
```

（5）打开注册页面 regist.aspx，在“注册”按钮注册成功的代码段，修改 commText 如代码清单 3-14 所示。

代码清单 3-14 修改后的注册代码

```
      string commText = "insert into userinfo (username,userpassw,realname,sex,
email,phone," +
                        "headpicture,[datetime],hobby,birthday,province,code)" +
    "values(@un,@up,@rn,@sex,@em,@ph,@hp,@dt,@hb,@bt,@pr,@cd)";
```

添加参数设置：

```
    //生成一个验证密码，这个语句放在插入数据库之前，code 与其他注册信息一起写入数据库
    string code=StringControl.getcode (txtName.Text,txtPassword.Text,txtEmail.
Text);
      sqlComm.Parameters.AddWithValue("@cd", code);
```

在插入成功后添加：

```
    //发送邮件到注册人信箱
    MailControl.sendMail (txtName.Text,txtEmail.Text,code);
```

（6）右击项目名，添加一个 Web 页面，文件名为 ActivateUser.aspx，该文件不设计任何界面，双击“设计”视图的空白处添加代码，参考代码如代码清单 3-15 所示。

代码清单 3-15 激活用户

```
    protected void Page_Load(object sender, EventArgs e)
        {
            if (!Page.IsPostBack)//假如页面是非回发方式访问
            {
                string result = "";
                //从邮箱的链接中获取用户名和验证码信息
                string username = Request.QueryString["username"].ToString();
                string code = Request.QueryString["Code"].ToString();
                //用于存放注册用户数据库中的验证码
                string codedata = "";
                SqlConnection sqlconn = new SqlConnection
    (ConfigurationManager.ConnectionStrings["addressconn"].ToString());
                //查找激活用户
                SqlCommand sqlcomm = new SqlCommand
    ("select code from userinfo where username=@uname and isactive=0", sqlconn);
                sqlcomm.Parameters.Add("@uname", SqlDbType.Char).Value = username;
                SqlDataReader sdr = null;
                try
                {
                    sqlconn.Open();
                    sdr = sqlcomm.ExecuteReader();
                    //如果该用户存在，则获取验证码
                    if (sdr.Read())
                    {
                        codedata = sdr.GetValue(0).ToString();
                    }
```

```
                    //假如验证码不存在，提示不存在信息
                    if (code == "")
                        result = @" 您好。 激活过程中发生错误，错误信息如下： "+
    "没有激活码，请点击邮箱中获得的激活链接来进行激活操作！";
                    else if (codedata.Trim() == code.Trim())
                    {
                        //如果验证码正确，把激活字段值设置为 1
                        sqlcomm = new SqlCommand
    ("update userinfo set isactive=1 where username=@uname", sqlconn);
                        sqlcomm.Parameters.Add("@uname", SqlDbType.VarChar).Value
= username;
                        if (sqlcomm.ExecuteNonQuery() > 0)
                            result = string.Format(@"{0}您好。 您已经成功完成了激活，"+
    "可以享受网络通讯录系统提供的各种服务了。", username);
                        else
                            result = string.Format(@"{0}您 好！激活过程中发生错误，错误
信息如下："+
    "已经过期的激活码，请点击邮箱中获得的激活链接来进行激活操作！", username);
                    }
                }
                catch { }
                finally
                {
                    if (sqlconn.State == ConnectionState.Open)
                        sqlconn.Close();
                    sdr.Close();
                }
                Response.Write(result);
            }
        }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-6 所示。

表 3-6　演练完成情况评价表

任　务　号	3-5	任 务 名 称	在 ASP.NET 程序中发送电子邮件
任务子项	完成情况	主要问题	未完成原因
添加注册码生成方法			
邮件发送类			
修改注册页面，在用户表写入激活码			
激活注册码			

三、知识点

在 System.Net.Mail 命名空间里包含用于将电子邮件发送到 SMTP 服务器的类。涉及发送邮件主要的类如下：

✧ MailMessage 类：设置邮件的内容；

✧ SmtpClient 类：将电子邮件传输到指定用于邮件传送的 SMTP 主机；

✧ Attachment 类：创建邮件附件；

✧ MailAddress 类：指定电子邮件发件人或收件人的地址。

1. MailMessage 类

MailMessage 类是.NET Framework 中管理邮件的专用类，表示可以使用 SmtpClient 类发送的电子邮件。该类提供了丰富的属性来创建电子邮件，它的构造函数有三个重载。下面的代码演示了如何创建 MailMessage 类的实例：

```
MailMessage message = new MailMessage ();     //无参数
//通过构造函数设置 SMTP 主机服务器
MailMessage message = new MailMessage ("smtp.Sina.com");
//通过构造函数设置 SMTP 主机服务器和端口
MailMessage message = new MailMessage ("smtp.Sina.com",25);
```

MailMessage 类的常用属性如下：

✧ AlternateViews 属性：获取用于存储邮件正文的替代形式的附件集合。使用 AlternateViews 属性指定一个电子邮件不同格式的副本，如果发送 HTML 格式的邮件，可能希望同时提供邮件的纯文本格式，以防一些收件人使用的电子邮件阅读程序无法显示 HTML 内容；

✧ Attachments 属性：获取用于存储附加到此电子邮件数据的附件集合；

✧ Bcc 属性：获取包含此电子邮件的密件抄送（BCC）收件人的地址集合。当收件人查看电子邮件时，通常不显示 BCC 地址；

✧ Body 属性：获取或设置邮件正文。如果正文内容还有其他替代格式以便为收件人提供更丰富的呈现选项，则可以使用 AlternateViews 属性指定正文内容的替代视图，如应用程序可能选择同时发送邮件正文的纯文本版本和 HTML 版本，可以显示 HTML 的电子邮件阅读程序可向收件人显示邮件的 HTML 版本，而无法显示 HTML 的阅读程序将显示邮件的纯文本版本；

✧ BodyEncoding 属性：获取或设置用于邮件正文的编码。为 BodyEncoding 属性指定的值设置 Content-Type 标头中的字符集字段；

✧ CC 属性：获取包含此电子邮件的抄送（CC）收件人的地址集合。若要将 CC 收件人添加到电子邮件中，应为该收件人地址创建一个 MailAddress，再将该对象添加到由 CC 属性返回的集合中，如：

```
MailAddress copy = new MailAddress ("Notification_List@contoso.com"); message.
CC.Add(copy);
```

✧ DeliveryNotificationOptions 属性：获取或设置此电子邮件的发送通知；

✧ From 属性：获取或设置此电子邮件的发信人地址；

✧ Headers 属性：获取与此电子邮件一起传输的电子邮件标头；

✧ IsBodyHtml 属性：获取或设置指示邮件正文是否为 HTML 格式的值；

✧ Priority 属性：获取或设置此电子邮件的优先级；

✧ ReplyTo 属性：获取或设置邮件的回复地址；

✧ Sender 属性：获取或设置此电子邮件的发件人地址，包含发件人地址信息的 MailAddress；

✧ Subject 属性：获取或设置此电子邮件的主题行；

✧ SubjectEncoding 属性：获取或设置此电子邮件的主题内容使用的编码；

✧ To 属性：获取包含此电子邮件收件人的地址集合。若要将收件人添加到电子邮件中，请为该收件人地址创建一个 MailAddress，再将该对象添加到由此属性返回的集合中，如果想给多人发送附件，可以将多人的收件地址连接在一起，中间通过分号“;”分隔。

2. SmtpClient 类

SmtpClient 类封装了将电子邮件发送到 SMTP 服务器的方法和属性。它用于让应用程序向 SMTP 服务器发送电子邮件，可以通过同步或异步的方法进行发送。通过结合 MailMessage 类使用，还可以设置邮件的格式、添加抄送人、添加附件等。下面的代码演示了如何创建 SmtpClient

类的实例：

```
    SmtpClient  client = new SmtpClient ("smtp.Sina.com");    //直接通过构造函数
设置 SMTP 主机服
    //务器
    //或：SmtpClient  client = new SmtpClient ();
    Client. Host ="smtp.Sina.com";    //通过 Host 属性来设置 SMTP 主机服务器
    //若要使用 SmtpClient 发送电子邮件，必须设置 Host、Port、Credentials 这三个属性
```

SmtpClient 类的常用属性和方法如下：

✧ Credentials 属性：获取或设置用于验证发件人身份的凭据；

✧ DeliveryMethod 属性：指定如何处理待发的电子邮件；

✧ EnableSsl 属性：指定 SmtpClient 是否使用安全套接字层（SSL）加密连接；

✧ Host 属性：获取或设置用于 SMTP 事务主机的名称或 IP 地址；

✧ PickupDirectoryLocation 属性：获取或设置文件夹，应用程序在该文件夹中保存将由本地 SMTP 服务器处理的邮件；

✧ Port 属性：获取或设置用于 SMTP 事务的端口；

✧ ServicePoint 属性：获取用于传输电子邮件的网络连接；

✧ Timeout 属性：获取或设置一个值，该值指定同步 Send 调用的超时时间；

✧ Send()方法：将指定的邮件发送到 SMTP 服务器以便进行传递。Send()方法有两个重载，Send(MailMessage message)和 Send(string from, string recipients, string subject, string body)。

3. Attachment 类

Attachment 类与 MailMessage 类结合在一起使用，用于给电子邮件添加附件。该类可以使用字符（String）和数据流（Stream）的形式创建附件，支持数据流的形式就意味着能用任何的文件格式作为附件，如.txt 格式或.doc 格式。Attachment 类的构造函数有 6 个重载，在这里不予一一介绍。典型的使用方式如下：

```
    Attachment data = new Attachment(textMessage);
    message.Attachments.Add(data);
```

Attachment 类的常用属性如下：

✧ ContentStream 属性：获取此附件的内容流；

✧ ContentType 属性：获取此附件的内容类型；

✧ TransferEncoding 属性：获取或设置此附件的编码。

4. MailAddress 类

MailAddress 类用于设置电子邮件发件人和收件人的地址。通过该类的属性可以获取电子邮件地址联系人的详细信息，如邮件中显示联系人的名字和 SMTP 服务器上的用户名。MailAddress 类的构造函数有三个重载，下面的代码是最常用的创建其实例的方式：

```
    //指定电子邮件的地址，构造一个新实例
    MailAddress FromMailBox = new MailAddress ("FromMailBox@Sina.com");
```

MailAddress 类的常用属性如下：

✧ Address 属性：获取电子邮件的地址；

✧ DisplayName 属性：获取电子邮件显示的名称；

✧ Host 属性：获取服务器名称，也就是在电子邮件地址@符号后的服务器名；

✧ User 属性：获取用户名称，也就是在电子邮件地址@符号前的用户名。

ASP.NET 就用 System.Net.Mail 命名空间下这四个类发送邮件。其中，SmtpClient 类发送电子邮件；MailMessage 类丰富电子邮件的内容；MailAddress 类设置电子邮件收件人和发件人的电子邮件地址；Attachment 类为邮件添加附件。

3.3.2 使用 Jmail 第三方组件实现邮件发送

使用 ASP.NET 邮件发送类实现 Web 项目中的邮件发送功能比较方便，但是要实现邮件接收功能是比较麻烦的，因此大部分程序在接收邮件时，基本使用第三方组件。

一、实战演练

（1）打开 Jmail 官方网站 http://www.dimac.net/，在“Free Downloads ”栏目下选择下载 w3Jmail 组件。

（2）Jmail 最新的压缩包中自带安装程序 jmail_free.msi，双击它可以进行自动安装，找到安装目录，可以看到 jmail.dll 组件。

（3）打开网络通讯录网站 address，在弹出的快捷菜单中选择 “添加”→“添加 ASP.NET 文件夹”→“Bin”。右击“Bin”目录，选择“添加”→“现有项”，找到 Jmail 组件安装目录，把 jmail.dll 组件添加到网站的 Bin 目录中。在“解决方案资源管理器”窗口中看到在 Bin 目录中增加了 jmail.dll 组件

（4）修改管理联系人页面，添加“发邮件”选项，修改后的管理联系人如图 3-7 所示。

（5）在“cpmanage”文件夹中添加 Jmailsend.aspx 文件，切换到“设计”视图，完成后的设计界面如图 3-8 所示。

图 3-7 修改后的管理联系人部分页面

图 3-8 Jmail 组件使用界面

（6）双击“发送”按钮，添加“发送”按钮的 Click 事件，事件代码如代码清单 3-16 所示。

代码清单 3-16 “发送”按钮的 Click 事件代码

```
//在使用前先添加 Jmail 类的命名空间
using jmail;
protected void submit_Click(object sender, EventArgs e)
    {
        jmail.MessageClass message = new MessageClass();  //添加一个邮件对象
        //设置邮件编码格式
        message.Charset = "gb2312";
        //设置发件信息：邮箱地址、姓名、主题及内容
        message.From = txtSendMail.Text;
        message.FromName= txtSendName.Text;
```

```
        message.Subject = txtSubject.Text;
        message.Body = txtContent.Text;
        //设置邮箱的用户名和密码
        message.MailServerUserName = txtUserName.Text;
        message.MailServerPassWord = txtPass.Text;
        //如果有附件，则添加附件一起上传
        if (FileUpload1.HasFile)
        {
            string filepath = FileUpload1.PostedFile.FileName;
            message.AddAttachment(filepath, true, "");
        }
        //设置收件人邮箱
        message.AddRecipient(txtAcceptMail.Text, "", "");
        try
        {
            message.Send(txtSmtp.Text, false);
            Label1.Text = "发送成功";
        }
        catch(Exception ex)
            {
                Label1.Text=ex.ToString();
            }
    }
```

（7）运行程序，输入信息并查看是否发送成功。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 3-7 所示。

表 3-7　演练完成情况评价表

任务号	3-6	任务名称	利用 Jmail 发送电子邮件
任务子项	完成情况	主要问题	未完成原因
添加引用			
发件箱界面设计			
实现发件功能			

三、知识点

Jmail 组件是由 Dimac 公司开发的用来完成邮件的发送、接收、加密、集群传输等工作的组件，它支持从 POP3 邮件服务器收取邮件，支持加密邮件的传输。

在使用之前先添加命名空间 using jmail，通过对象浏览器可以查看 Jmail 中所有的类及类的方法与属性。下面介绍几个常用的类。

1. POP3 对象

用于建立接收邮件，继承于 POP3 的常用类是 POP3Class，实例化格式为：

```
POP3Class popMail = new POP3Class();
```

POP3Class 类常用的属性和方法如下：

✧ Count 属性：获取 POP3 服务器上的邮件数目；

✧ Loggin 属性：设置或获取能否打开日志功能，默认为关闭；

✧ Log 属性：获取当 Logging 被设置为 true 时，w3 Jmail 创建的日志；

✧ Size 属性：获取邮箱的总字节数；

✧ Messages 属性：获取一个可以被读写的 Messages 对象；

✧ Timeout 属性：设置或获取 POP3 服务器连接超时时间，单位是秒；

✧ Connect()方法：连接指定的 POP3 服务器，基本格式为 Connect(string Username,string Password,string Server,int Port)，其中端口号可选，默认为 110；

✧ DeleteMessages()方法：从邮件服务器上删除所有邮件；

✧ DeleteSingleMessage(MessageID) 方法：从邮件服务器上删除由 MessageID 指定的邮件；

✧ Disconnect()方法：关闭和邮件服务器的连接；

✧ DownloadHeaders()方法：从邮件服务器上读取所有的邮件头并传递给 Messages 集合；

✧ DownloadMessages()方法：从邮件服务器上读取所有邮件；

✧ DownloadSingleHeader(MessageID)方法：从邮件服务器上读取指定的邮件头并传递给 Messages 集合；

✧ DownloadUnreadMessages()方法：从邮件服务器上读取所有未读邮件。

2. Message 对象

用于建立邮件信息，MessageClass 类是继承 Message 最常用的类，其作用相当于 ASP.NET 中的 MailMessage 类。

Messageclass 类常用的属性和方法如下：

✧ About 属性：设置一些附加信息；

✧ Attachments 属性：返回邮件的附件集合，Set Attachments=Message.Attachments；

✧ Body 属性：返回邮件正文；

✧ BodyText 属性：返回全部的文本正文；

✧ Charset 属性：设置邮件使用的字符集，默认为 US-ASCII，支持中文时则设置为 GB 2312；

✧ Date 属性：返回邮件的发送时间；

✧ DeferredDelivery 属性：设置邮件定时发送；

✧ Encoding 属性：设置附件的默认编码 Base64 或 Quoted-Printable ；

✧ From 属性：返回或设置发件人的 E-mail 地址；

✧ FromName 属性：返回或设置发件人的名称；

✧ HtmlBody 属性：返回或设置邮件正文的 HTML 部分；

✧ Log 属性：当 Logging 为 true 时，该函数返回创建的日志；

✧ Logging 属性：是否启用日志，Message.Logging 为 true 时，启用日志；

✧ MailServerPassWord 属性：当邮件服务器使用 SMTP 发信认证时，该函数设置登录密码；

✧ MailServerUserName 属性：当邮件服务器使用 SMTP 发信认证时，该函数设置登录账号；

✧ MsPickupdirectory 属性：指定 SMTP 服务器 Pickup 文件夹的位置；

✧ Priority 属性：设置邮件的优先级 1、2 和 3 ；

✧ Recipients 属性：返回收件人集合；

✧ RecipientsString 属性：返回收件人集合（只读）；

✧ ReplyTo 属性：指定一个回复地址；

✧ Size 属性：返回邮件的总字节数；

✧ Subject 属性：设置邮件标题；

✧ Text 属性：返回完整的邮件内容；

✧ AddAttachment()方法：给邮件添加一个文件型的附件，当其中的参数 isInline 设置为 true 时，添加的附件就是一个可嵌入的附件；

✧ AddCustomAttachment()方法：给邮件添加一个自定义类型的附件；

✧ AddHeader()方法：给邮件添加一个自定义邮件头 X-Header；

✧ AddNativeHeader()方法：给邮件添加一个邮件头；

✧ AddRecipient()方法：给邮件添加一个收件人，参数 RecipientName 和 PGPKey 是可选项，RecipientName 为收件人姓名，用 PGPKey 给邮件加密；

✧ AddRecipientBCC()方法：添加一个邮件暗送人地址；

✧ AddRecipientCC()方法：给邮件添加一个邮件抄送人地址；

✧ AddURLAttachment()方法：从指定的 URL 下载文件并添加为邮件附件，参数 bstrAttachAs 是用来更改添加邮件附件的文件名；

✧ AppendBodyFromFile()方法：清除邮件正文，并把指定文件的内容作为邮件正文；

✧ AppendHTML()方法：从邮件追加 HTML 格式正文，如 Message. AppendHTML(“<H3>Hello Word</H3>”)；

✧ AppendText()方法：给邮件添加文本正文；

✧ Clear()方法：清除所有的邮件消息，使之成为一个空对象；

✧ ClearAttachments()方法：清除附件列表；

✧ ClearCustomHeaders()方法：清除所有自定义的邮件头；

✧ ClearRecipients()方法：清除所有收件人的地址列表；

✧ Close()方法：释放 Jmail 与邮件服务器连接而使用的缓存；

✧ DecodeHeader()方法：输出一个邮件头消息；

✧ ExtractEmailAddressesFromURL()方法：从指定的网址（URL）读取并添加邮件列表；

✧ Nq()方法：将邮件追加到发送队列等待发送；

✧ ParseMessage()方法：解析一个邮件，数据流必须符合 RFC822 格式标准；

✧ SaveToStream()方法：保存邮件到数据流，数据流必须符合 RFC822 格式标准；

✧ Send()方法：发送邮件。邮件服务器是一个描述邮件服务器名称或地址的字符串，用户名和密码是可选项，当要发送认证邮件时使用的格式为用户名:密码@邮件服务器；

✧ SendToNewsGroup()方法：使用指定的 SMTP 服务器发送邮件到新闻组，多个邮件用“,”隔开。

3. Messages 对象

Messages 对象是一个 Message 对象的集合，Messages 的继承类是 MessagesClass，典型使用如下：

```
Messages mails=popMail.Messages;
```

Messages 对象常用的属性和方法如下：

✧ Count 属性：返回集合中记录的数目；

✧ Clear()方法：清除集合中的所有内容，并不会删除邮件服务器上的任何邮件。

4. Attachment 对象

Attachment 对象用于给电子邮件添加附件。

Attachment 对象常用的属性和方法如下：

✧ New()：创建一个可以加入到 Attachments 集合的附件，如果指定 Data 的值，那么 Jmail 将创建一个以 Data 参数为内容的自定义附件；

✧ SaveToFile()：保存附件到硬盘；

✧ ContentType()：返回附件类型；

✧ Data()：返回附件的内容；

✧ BinaryData()：以二进制模式返回附件的内容；

✧ IsInline()：如果附件被设置为可嵌入（inline）的，则返回 true；

✧ Name()：返回附件的文件名称；

✧ Size()：返回附件的大小。

5. Attachments 对象

Attachments 对象是一个 Attachment 对象的集合。

Attachments 对象常用的属性和方法如下：

✧ Add(Attachment)：添加一个附件到集合；

✧ Clear()：清除集合中的全部附件；

✧ Count()：返回集合中附件的总数。

四、任务拓展

学习本节后要求完成一个课外拓展实践任务。

拓展任务卡 3

拓展任务号	3-3	任务名称	使用 Jmail 组件接收邮件
计划用时	45 分钟	任务性质	课外
任务描述与目标			
Jmail 组件的强大功能可以实现邮件的接收			
主要操作步骤提示			

1. 打开 mails 项目，添加一个新的 Web 页面，文件命名为 AcceptMail.aspx；

2. 在“设计”视图中，设计如图 3-9 所示的接收邮件界面；

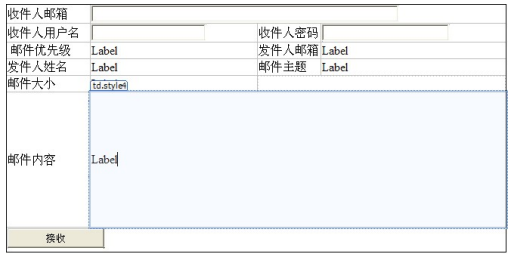

图 3-9 接收邮件界面

3. 接收邮件的参考代码如下：

```
protected void Button1_Click(object sender, EventArgs e)
 {
 POP3Class popMail = new POP3Class();        //建立收邮件对象
Message mailMessage;                          //建立邮件信息接口
Attachments atts;                            //建立附件集接口
Attachment att;                              //建立附件接口
```

续表

拓展任务号	3-3	任务名称	使用 Jmail 组件接收邮件
计划用时	45 分钟	任务性质	课外

```
try
{
popMail.Connect(txtUserName.Text, txtPass.Text, txtAccept.Text, 110); //连接到 POP3 服务器
 if (popMail.Count > 0)     //如果收到邮件
    {
      for (int i = 1; i <2; i++)  //获取第一条邮件
        {
         mailMessage = popMail.Messages[i];            //取得一条邮件信息
          atts = mailMessage.Attachments;              //取得该邮件的附件集合
           mailMessage.Charset = "GB 2312";            //设置邮件及附件的编码
   mailMessage.Encoding = "Base64";
          mailMessage.ISOEncodeHeaders = false;       //是否将信头编码为 ISO-8859-1 字符集
          LbPriority.Text = mailMessage.Priority.ToString();//邮件的优先级
          Lbform.Text = mailMessage.From;             //邮件发送人的信箱地址
          LbtFromname.Text = mailMessage.FromName;    //邮件的发送人
          LbSubject.Text = mailMessage.Subject;       //邮件主题
          LbContent.Text = mailMessage.Body;          //邮件内容
          Lbsize.Text = mailMessage.Size.ToString(); //邮件大小
          att = atts[0]; //取得附件
  string attname = att.Name; //附件名称
          att.SaveToFile(Server.MapPath(@"~/Files/") + attname); //上传到服务器
        }
      att = null;  atts = null;
      }
     else
       {            Response.Write("没有新邮件!");    }
  popMail.Disconnect();  //关闭与邮件服务器的连接
          popMail = null;
      }
      catch (Exception ex)
      {
          Response.Write(ex);
          Response.Write("请检查邮件服务器的设置是否正确！");
      }
   }
```

拓展任务跟踪卡

任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

项目号			项目名	
项目完成方式		□小组协作 □个人独立		
项目完成情况（60%）	代码编写（30%）	自我评价		
		小组评价		
		个人评价		
	功能实现（20%）	自我评价		
		小组评价		
		个人评价		
	说明文档（10%）	自我评价		
		小组评价		
		个人评价		
拓展与创新（40%）	创新能力（20%）	自我评价		
		小组评价		
		个人评价		
	设计潜力（20%）	自我评价		
		小组评价		
主要存在问题				

课外思考题

1．在 Web 开发中为什么要增加验证码功能？目前各类论坛中经常出现帖子“灌水”的现象，请您设计一个方案，避免这个现象。

2．在.NET 平台中随机数是由 Random 类来生成的，但是该类不是真正的随机数生成类，而称为伪随机数生成器，请问为什么？如果想生成十组 10～100 不同的随机数，请问应该如何实现？

3．现在很多小说网上，最新章节的内容问题是以图片的形式显示在网页上的，而作者是以文本的方式上传的，请写出设计流程，如果可以请写出实现代码。

项目四

信 息 处 理

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 掌握.NET 中文件系统的管理；
- ✧ 掌握 ASP.NET 文件上传与显示的基本原理；
- ✧ 掌握 ASP.NET 中图片的存储方式与图片的处理方法；
- ✧ 掌握 ASP.NET 文本信息处理方式；
- ✧ 了解第三方组件在 ASP.NET 中的应用。

技能目标

通过对本项目的学习，应该具备下面的能力：

- ✧ 能根据网站项目情况实现文件的上传与下载功能；
- ✧ 能选择合理的方法存储网站中的图片数据；
- ✧ 能选择合适的方式实现在线文本编辑功能。

教学建议

本项目计划总学时为 8 学时。

- ✧ 情境介绍：1 学时；
- ✧ 任务 1：2 学时；
- ✧ 任务 2：2 学时；
- ✧ 任务 3：3 学时。

经过之前三个项目的学习，学生对网站开发有了一定的认识。教师可以用一个多行文本框来上传文本，但是上传后的文本没有办法有格式地进行显示，来引发本项目的学习。

4.1 情境介绍

信息是网络交流的一种重要媒体，将数据信息放在网络上共享，是网络资源共享的常用方式。网络信息包括文字信息、图片信息及各类网络共享文件等多方面的信息。信息处理是网站项目开发要完成的重要功能。

在网站开发中文件处理主要涉及文件的上传、下载和删除，是在实际项目开发过程中经常需要用到的技术。上传是将文件按照一定的格式和规范，放到服务器指定的目录，同时为了保障网站服务器的安全，上传文件时可以限定上传文件的类型、大小。下载是将网站服务器上的共享文件保存到浏览者的计算机中。

文字处理主要是指利用一些编辑功能编辑文章，同时页面显示结果与编辑时显示的效果一致，ASP.NET 所提供的控件远远不能达到所需的要求。要实现这个功能可以使用创建自定义控件方式，也可以使用第三方控件来完成。

在实际的应用中往往是要求文字处理与文件上传同步进行。为实现图文信息同时存放，图片数据存储表 addimagedata 的设计如图 4-1 所示。

列名	数据类型	允许空
imageID	int	☐
imagename	varchar(50)	☐
imagedata	image	☐
imagepath	nvarchar(500)	☑
imageintro	text	☑

图 4-1 图片数据存储表 addimagedata 的设计

4.2 任务 1 ASP.NET 文件处理与上下文信息

文件上传在实际的处理中可以是单文件的保存方式，这种保存方式不通过数据库处理，本任务主要完成文件的上传与下载，学习如何让浏览用户把文件上传到服务器指定的目录中。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）学习.NET 常用的文件系统管理类；

（2）实现单文件上传；

（3）实现批量文件上传；

（4）实现文件下载功能。

4.2.1 ASP.NET 文件处理

一、实战演练

（1）打开 Visual Studio 2012，新建一个站点 infoAccess。

（2）新建一个 Web 页面“singlefile.aspx”，设计一个上传功能的界面，如图 4-2 所示。

图 4-2 上传功能的界面设计

（3）右击项目名 infoAccess，添加一个文件夹“uploadfile”，用于存入上传的文件。

（4）双击“上传”按钮，添加按钮的 Click 事件，添加上

传代码，如代码清单 4-1 所示。

代码清单 4-1 “上传”按钮的 Click 事件代码及上传代码

```
protected void Btnupload_Click(object sender, EventArgs e)
    {
        bool uploadflag = false;
        //设置上传文件存放的服务器路径
        string filepath = Server.MapPath("~/uploadfile/");
        //设置可上传文件的类型
        string[] filetype = {".txt",".doc",".jpg",".gif",".xls" };
        if (FileUpload1.HasFile)
        {
            //查看上传的文件类型是否符合要求
           string fileExt = Path.GetExtension(FileUpload1.FileName).ToLower();
            for (int i = 0; i < filetype.Length; i++)
            {
                if (fileExt == filetype[i])
                    uploadflag = true;
            }
            //文件大小控制在 2MB 以内
            if (FileUpload1.FileBytes.Length > 2 * 1024 * 1024)
                uploadflag = false;
            //检查上传文件的文件名是否已存在
            DirectoryInfo Dirfile = new DirectoryInfo(filepath);
            FileInfo[] files = Dirfile.GetFiles();
            for (int i = 0; i <files.Length; i++)
            {
                if (FileUpload1.FileName == files[i].Name)
                    uploadflag = false;
            }
            if (uploadflag)
            {
                try
                {
                    FileUpload1.SaveAs(filepath + FileUpload1.FileName);
                    Label1.Text = "上传成功";
                    HyperLink1.Visible = true;
                }
                catch
                {
                    Label1.Text = "文件无法实现上传";
                }
            }
            else
                Label1.Text = "文件太大、类型不对或该文件名已经存在，不能上传";
        }
        else
            Label1.Text="请先选择上传的文件";
    }
```

（5）打开 singlefile.aspx 文件，运行调试。上传一个不符合要求的文件，页面显示如图 4-3 所示的结果；上传一个符合要求的文件后，在页面中显示“单击查看上传文件”字样，如图 4-4 所示。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，演练完成情况评价表如表 4-1 所示。

图 4-3 上传文件有错

图 4-4 上传成功后的页面

表 4-1 演练完成情况评价表

任务号	4-1	任务名称	文件上传
完成情况		主要问题	未完成原因

三、知识点

大多数 Web 应用程序都允许用户上传文件，并在一定权限范围内允许用户创建、修改和保存文件。.NET 平台提供了管理驱动器、目录和文件的类。

1. 文件路径访问管理 Path 类

每个驱动器包含一个或多个目录，而每个子目录又可以包含一个或多个目录和文件，从而构成了树状的目录结构。通过 Path 路径访问管理类，用户可以访问存储在树状结构中的文件和目录路径的每个段，包括驱动器盘符、目录名、文件名、文件扩展名及路径分隔符。Path 类是静态共享类，使用时不需要经过实例化。

其常用属性和方法如下。

✧ DirectorySeparatorChar 属性：提供平台特定的字符，该字符用于在反映分层文件系统组织的路径字符串中分隔目录级别，在 Windows 操作系统中为反斜杠（\）；

✧ PathSeparator 属性：用于在环境变量中分隔路径字符串的平台特定的分隔符，在基于 Windows 的桌面平台上，默认情况下该字段的值是分号（;）；

✧ GetDirectoryName()方法：返回指定路径字符串的目录信息，包括驱动盘符和文件名的路径。“C:\mydir\myfile.ext”返回“C:\mydir”，如果路径由根目录组成，如“c:\”则返回 null；

✧ GetExtension()方法：返回某个指定文件的路径中文件的扩展名。如果 path 为 null 引用，则此方法返回 null 引用；如果 path 不具有扩展名信息，则此方法返回 Empty；

✧ GetFileName()方法：返回指定路径字符串的文件名和扩展名，由 path 中最后的目录字符后的字符组成。如果 path 的最后一个字符是目录或卷分隔符，则此方法返回 Empty；如果 path 为 null 引用，则此方法返回 null 引用；

✧ GetFullPath()方法：返回指定路径字符串的绝对路径，如“C:\MyFile.txt”；

✧ HasExtension 方法：确定路径是否包括文件扩展名。从 path 的结尾开始，搜索后面跟有至少一个字符的句点（.）；

2. 使用 File 和 FileInfo 类访问文件

File 类提供用于创建、复制、删除、移动和打开文件的静态方法；FileInfo 类提供用于创建、复制、删除、移动和打开文件的实例方法。二者都协助创建 FileStream。

1）File 类

将 File 类用于如复制、移动、重命名、创建、打开、删除和追加到文件的典型操作，由于所有的 File 方法都是静态的，所以如果只想执行一个操作，那么使用 File 方法的效率比使用相应的 FileInfo 实例方法可能更高。File 类的静态方法对所有方法都执行安全检查，所有的方法都是对当前指定的文件路径。File 类的常用方法如下。

✧ Copy()方法：将现有文件复制到新文件，如：File.Copy(path, path2)；

✧ Create()方法：在指定路径中创建文件，如：File.Create(@"c:\temp\MyTest.txt")；

✧ Delete()方法：删除指定的文件，如果指定的文件不存在，不引发异常，如：File.Delete(@"c:\temp\MyTest.txt")；

✧ Exists()方法：确定指定的文件是否存在，如：File.Exists(@"c:\temp\MyTest.txt")；

✧ Move()方法：将指定文件移到新位置，并提供指定新文件名的选项，如：File.Move(path, path2)；

✧ Open()方法：打开指定路径上的 FileStream，如 FileStream fs=File.Open(path,FileMode.Open, FileAccess.Write)；

✧ ReadAllText()方法：打开一个文本文件，将文件的所有行读入一个字符串，然后关闭该文件；

✧ WriteAllText()方法：创建一个新文件，在文件中写入内容，然后关闭文件，如果目标文件已存在，则覆盖该文件；

2）FileInfo 类

如果打算多次重用某个对象，用 FileInfo 的相应实例方法效率会更高些，FileInfo 类仅在实例化时执行一次用户安全检查。FileInfo 类的常用属性和方法如下。

✧ Directory 属性：获取包含文件的目录实例；

✧ DirectoryName 属性：获取表示目录完整路径的字符串；

✧ Exists 属性：获取指示文件是否存在的值；

✧ Extension 属性：获取表示文件扩展名部分的字符串；

✧ FullName 属性：获取目录或文件的完整目录；

✧ Length 属性：获取当前文件的大小（字节）；

✧ Name 属性：获取文件名；

✧ AppendText()方法：创建一个 StreamWriter，它向 FileInfo 实例表示的文件追加文本；

✧ CopyTo()方法：将现有文件复制到新文件；

✧ Create()方法：创建文件；

✧ Delete()方法：永久删除文件；

✧ Open()方法：用各种读/写访问权限和共享特权打开文件。

下面的代码清单 4-2 展示了类的一些常见操作。

代码清单 4-2 FileInfo 类的基本用法

```
    string path = @"c:\MyTest.txt";
    FileInfo fi = new FileInfo(path);
        // 假如文件已经存在先删除
      if (fi.Exists)
      {
         fi.Delete();
      }
        //创建文件.
      using (FileStream fs = fi.Create())
        {
           Byte[] info = new UTF8Encoding(true).GetBytes("This is some text in
the file.");
    //Add some information to the file.
            fs.Write(info, 0, info.Length);
        }
           //打开文件并读取显示在页面上.
       using (StreamReader sr = fi.OpenText())
       {
            string s = "";
           while ((s = sr.ReadLine()) != null)
           {
                  textBox1.Text=s;
           }
        }
```

3. 使用 Directory 和 DirectoryInfo 类访问目录

.NET 在 System.IO 命名空间中提供了用户操作目录的 2 个类：Directory 和 DirectoryInfo 类。Directory 类提供创建、移动和枚举目录和子目录的静态方法。DirectoryInfo 提供创建、移动和枚举目录和子目录的实例方法。

1）Directory 类

由于所有的 Directory 方法都是静态的，所以如果只想执行一个操作，那么使用 Directory 方法的效率比使用相应的 DirectoryInfo 实例方法可能更高。大多数 Directory 方法要求当前操作的目录的路径。**Directory** 类用于典型操作，如复制、移动、重命名、创建和删除目录的常用方法如下：

✧ CreateDirectory()方法：创建 path 中指定目录，除非这些目录已存在或 path 的某一部分无效，path 参数指定目录路径，而不是文件路径，如果目录已经存在，则此方法不执行任何操作，如：Directory.CreateDirectory (@"c:\MyDir")；

✧ Delete()方法：删除指定的目录，如：Directory.Delete (@"c:\MyDir");

✧ Exists()方法：确定给定路径是否引用磁盘上的现有目录；

✧ GetDirectories()方法：获取指定目录中子目录的名称，返回值为 String[]，使用如 string[] subdirectory= Directory.GetDirectories(@"c:\MyDir")；

✧ GetFiles()方法：返回指定目录中文件的名称，返回值为 String[]，使用如 string[] Files= Directory. GetFiles (@"c:\MyDir")；

✧ GetParent()方法：检索指定路径的父目录，包括绝对路径和相对路径；

✧ Move()方法：将文件或目录及其内容移到新位置，使用如 Directory.Move("C:\\proof", "C:\\Temp")。

2）DirectoryInfo 类

如果对目录打算多次操作，则使用 DirectoryInfo 的实例方法比较好，DirectoryInfo 类的方法和属性方法如下。

✧ Exists 属性：获取指示目录是否存在的值；

✧ Extension 属性：获取表示文件扩展名部分的字符串；

✧ FullName 属性：获取目录或文件的完整目录；

✧ Name 属性：获取此 DirectoryInfo 实例的名称；

✧ Parent 属性：获取指定子目录的父目录；

✧ Root 属性：获取路径的根部分；

✧ CreateSubdirectory()方法：在指定路径中创建子目录；

✧ Delete()方法：从路径中删除目录及其内容；

✧ GetDirectories()方法：返回当前目录的子目录；

✧ GetFiles()方法：返回当前目录的文件列表。

下面的代码展示了 DirectoryInfo 类的一些常见操作。

```
DirectoryInfo di = new DirectoryInfo(@"c:\MyDir");
if (!di.Exists)
{ di.Create();}
else
{ DirectoryInfo[] diArr = di.GetDirectories();
FileInfo[] fiArr = di.GetFiles();}
```

4. 使用 StreamReader 和 StreamWriter 读写文件

Stream 类用于字节的输入和输出，而 StreamReader /StreamWriter 可以对文本文件信息进行读写。StreamReader 和 StreamWriter 的默认编码为 UTF-8，UTF-8 可以正确处理 Unicode 字符。一般的 ASP.NET 字符串以 Unicode 进行编码。如代码清单 4-3 演示了文本文件的读写。

代码清单 4-3　演示文本文件的读写

```
string path = @"c:\temp.txt";
       if (!File.Exists(path))
       {          File.Create(path);       }
       StreamWriter sw = new StreamWriter(path);
       sw.Write("This is the ");
       sw.WriteLine("header for the file.");
       sw.Close();
       StreamReader sr = new StreamReader(path);
       String line;
       while ((line = sr.ReadLine()) != null)
       {
          Response.Write(line);
       }
       sr.Close();
```

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 1

拓展任务号	4-1	任务名称	添加文件目录
计划用时	30 分钟	任务性质	课内
任务描述与目标			
在对网站文件的管理中，如果所有的文件都存放在一起，会显示杂乱无章，因此在对文件的管理中，也涉及了目录的管理			
主要操作步骤提示			

1．页面中添加一个文本框，一个添加目录的功能按钮，一个标签；
2．在新添加的按钮中编写添加目录功能，参考代码如下：

```
  bool  dirFlag = false;
string filepath = Server.MapPath("~/uploadfile/");
  DirectoryInfo Dirfile = new DirectoryInfo(filepath);
  DirectoryInfo [] dirs = Dirfile. GetDirectories() ;
  for (int i = 0; i <dirs.Length; i++)
     {
        if (txtDirName.Text== dirs[i].Name)
           dirFlag = false;
     }
if(!dirFlag)  LabelDir.Text="目录已存在，请重新输入"
else
Dirfile.CreateSubdirectory(txtDirName.Text);
```

4.2.2 ASP.NET 上下文信息

一、实战演练

（1）添加一个 Web 页面“multifile.aspx”，设计一个多文件上传功能界面，如图 4-5 所示。

（2）添加“增加”按钮的客户端事件，<input type="button" value="增加" onclick="addFile()">，动态添加文件上传标记，如代码清单 4-4 所示。

代码清单 4-4 “增加”按钮的客户端事件代码

```
    <script    language="JavaScript">
          function    addFile()
         {
           var str ='<br/><INPUT    type="file"    size="50"    NAME="File">'

document.getElementById('MyFile').insertAdjacentHTML("beforeEnd",str)
          }
    </script>
```

（3）运行程序调度，单击页面中的“增加”按钮，添加文件上传控件，运行后的界面如图 4-6 所示。

（4）添加“开始上传”按钮事件 UploadButton_Click，实现多文件上传功能，如代码清单 4-5 所示。

图 4-5　多文件上传功能界面

图 4-6　多文件上传的运行界面

代码清单 4-5　“开始上传”按钮事件 UploadButton_Click 代码

```
protected void UploadButton_Click(object sender, EventArgs e)
    {
        SaveImages();
    }
    private void SaveImages()
    {
        // 遍历 File 表单元素
        HttpFileCollection files = HttpContext.Current.Request.Files;
        // 状态信息
        System.Text.StringBuilder strMsg = new System.Text.StringBuilder();
        strMsg.Append("<br><br><br>上传的文件分别是：<hr     color=red><br>");
        try
        {
            for (int iFile = 0; iFile < files.Count; iFile++)
            {
                //检查文件扩展名字
                HttpPostedFile postedFile = files[iFile];
                string fileName;
                fileName =Path.GetFileName(postedFile.FileName);
                if (fileName != "")
                {
                    strMsg.Append(fileName + "<br>");
                 postedFile.SaveAs(Server.MapPath("~/uploadfile/") + fileName);
                }
            }
            showResult.Text = strMsg.ToString();
        }
        catch (System.Exception Ex)
        {
            showResult.Text = Ex.Message;

        }
    }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-2 所示。

表 4-2　演练完成情况评价表

任务号	4-2	任务名称	实现多文件上传
任务子项	完成情况	主要问题	未完成原因
客户端事件			
上传多文件事件			

三、知识点

HttpContext 类封装了有关 HTTP 请求的所有 HTTP 特定的信息，也叫上下文信息。该类从客户端用户单击并产生了一个向服务器发送请求开始，到服务器处理完请求并生成返回到客户端为止，针对每个不同用户的请求，服务器都会创建一个新的 HttpContext 实例直到请求结束，服务器销毁这个实例。该类对 Request、Respose、Server 等都进行了封装，并保证在整个请求周期内都可以随时随地地调用。

其常用属性如下。

✧ Cache 属性：获取当前应用程序域的 Cache 对象；

✧ Current 属性：为当前 HTTP 请求获取或设置 HttpContext 对象；

✧ Request 属性：为当前 HTTP 请求获取 HttpRequest 对象；

✧ Response 属性：为当前 HTTP 请求获取 HttpResponse 对象；

✧ Server 属性：获取提供用于处理 Web 请求方法的 HttpServerUtility 对象；

✧ Session 属性：为当前 HTTP 请求获取 HttpSessionState 对象。

4.2.3 ASP.NET 常用编码格式

文件上传后，浏览者能查看到上传的文件信息，并通过单击文件名进行文件下载。

一、实战演练

（1）在项目中添加“showfiles.aspx”Web 页面，在页面中添加一个显示文件列表的 Panel 控件，在控件上方添加提示信息“文件显示清单”，添加 Page_Load 事件，如代码清单 4-6 所示。

代码清单 4-6 文件列表显示页面的 Page_Load 事件源代码

```
protected void Page_Load(object sender, EventArgs e)
{
    bool filetype = false;
    string filepath = Server.MapPath("~/uploadfile/");
    //创建目录对象
    DirectoryInfo Dirfile = new DirectoryInfo(filepath);
    //获取存放文件目录中所有文件的信息
    FileInfo[] files = Dirfile.GetFiles();
    for (int i = 0; i < files.Length; i++)
    {
        //过滤掉上传文件类型以外的文件
        string[] filetypes = { ".txt", ".doc", ".jpg", ".gif", ".xls" };
       string fileExt = Path.GetExtension(filepath + files[i].Name).ToLower();
        for (int j= 0; j < filetypes.Length; j++)
        {
            if (fileExt== filetypes[j])
            {
                filetype = true;
                break;
            }
        }
        //显示文件，并可以通过单击文件名转到下载页面进行文件的下载
        if (filetype)
        {
          Panel1.Controls.Add(new    LiteralControl("<a    href='download.aspx?
fname=" +  files[i]. Name + "'>" + files[i].Name + "</a>"));
```

```
            Panel1.Controls.Add(new LiteralControl("<br/>"));
        }
        filetype =false;
    }
}
```

（2）在项目中添加下载文件的 Web 页面“download.aspx”，页面中不添加任何控件。在 Page_Load 事件添加实现下载功能代码，如代码清单 4-7 所示。

代码清单 4-7　下载文件页面的 Page_Load 事件源代码

```
protected void Page_Load(object sender, EventArgs e)
    {
string filepath = Server.MapPath("~/uploadfile/");
//获取文件清单页面传递过来的文件名
        string selectname = Request.QueryString["fname"];
        //调用下载文件的方法
          savefile(filepath, selectname);
    }
    void savefile(string filepath, string filename)
    {
        string filefull = filepath + filename;
        FileInfo file = new FileInfo(filefull);
        Response.Clear();
        Response.Charset = "utf-8";
        Response.Buffer = true;
//关闭 ViewState 以提高速度
        this.EnableViewState = false;
        //定义输出文件编码、类型及文件名
        Response.ContentEncoding = System.Text.Encoding.UTF8;
        Response.HeaderEncoding = System.Text.Encoding.UTF32;
       Response.AppendHeader("Content-Disposition", "attachment; filename=" +
HttpUtility.UrlEncode(filename.ToString()));
        //保存文件类型不限，因此选择“unknow”
        Response.ContentType = "application/unknow";
        Response.WriteFile(filefull);
        Response.Flush();
        Response.Close();
        Response.End();
    }
```

（3）运行“showfiles.aspx”Web 页面，进入如图 4-7 所示的文件列表显示页面。选择某个文件并单击，弹出如图 4-8 所示的“文件下载”对话框，在对话框中单击“保存”按钮，在弹出的“文件保存”对话框中保存文件到指定的目录。

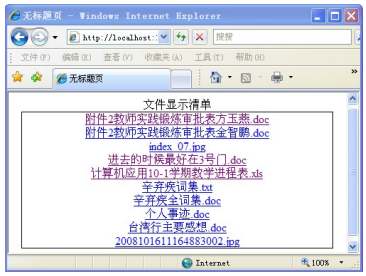

图 4-7　文件列表显示页面

图 4-8　“文件下载”对话框

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-3 所示。

表 4-3 演练完成情况评价表

任 务 号	4-3	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
显示文件列表页面			
下载文件页面			

三、知识点

1. 字符集与编码

字符（Character）是各种文字和符号的总称，包括各国家的文字、标点符号、图形符号、数字等。字符集是多个字符的集合，字符集种类较多，每个字符集包含的字符个数不同，常见的字符集有：ASCII、GB 2312、Unicode、ISO 8859、UTF 等。

计算机要准确地处理各种字符集文字，需要进行字符编码，以便能够识别和存储各种文字。制定编码首先要确定字符集，并将字符集内的字符排序，然后和二进制数字对应起来。根据字符集内字符的多少，会确定用几个字节来编码。

Unicode 是一种在计算机上使用的字符编码，它是 http://www.unicode.org 制定的编码机制，力求将全世界常用文字都函括进去。它为每种语言中的每个字符设定了统一并且唯一的二进制编码，以满足跨语言、跨平台进行文本转换、处理的要求。

一个字符的 Unicode 编码是确定的，但是在实际传输过程中，由于不同系统平台的设计不一定一致，以及出于节省空间的目的，对 Unicode 编码的实现方式有所不同。

Unicode 的实现方式称为 Unicode 转换格式（Universal Transformation Format， UTF）。

✧ UTF-8：8 bit 变长编码，对于大多数常用字符集（ASCII 中 0～127 字符），它只使用单字节，而对其他常用字符（特别是朝鲜和汉语会意文字），它使用 3 字节；

✧ UTF-16：16 bit 编码，是变长码，大致相当于 20 位编码，值在 0～0×10FFFF，基本上就是 Unicode 编码的实现；

✧ UTF-32：32 bit 编码，仅使用了 Unicode 范围（0～0×10FFFF）的 32 位编码。

UTF-16、UTF-32 有一个缺点就是浪费空间，如 ASCII 字符集中的内容完全没有必要使用 2 个甚至 4 个字节来存储，而 UTF-8 就解决了节省空间的问题。

字符在保存时的编码格式如果和显示的编码格式不一样，就会出现乱码问题。在 Web 系统中，从底层数据库编码、Web 应用程序编码到 HTML 页面编码，如果有一项不一致，就会出现乱码。所以，解决乱码问题的关键是让交互系统之间的编码一致。

2. Encoding 类

Encoding 类主要用于在不同的编码和 Unicode 之间进行转换，属于 System.Text 命名空间。其常见的属性和方法如下。

✧ Unicode 属性：获取 UTF-16 格式的编码；

✧ UTF32 属性：获取 UTF-32 格式的编码；

✧ UTF8 属性：获取 UTF-8 格式的编码；

✧ ASCII 属性：获取 ASCII（7 位）字符集的编码；

✧ GetEncodings()方法：返回包含所有编码的 EncodingInfo 类型的数组，可以通过 EncodingInfo 类的 CodePage 属性和 Name 属性查看编码的代码页标识符和名称。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 2

拓展任务号	4-2	任务名称	编写文件管理类
计划用时	30 分钟	任务性质	课内
任务描述与目标			
在对网站文件的管理中，如果对不用的文件不进行删除，那么时间久了会产生一些垃圾文件，因此应该对一些不用的文件及时进行清理			
主要操作步骤提示			

1．右击项目，添加一个文件管理类 filemanage.cs；
2．在类文件中，把 download.aspx.cs 中的 savefile()方法剪切到类中，并把方法访问属性改为 public，同时适当修改页面中的 Page_Load 事件代码；
3．在类中添加一个 deletefile()方法，参考代码如下：
`public void deletefile(string filepath,string filename)`
`{`
`string filefull = filepath + filename;`
`File.Delete(filefull);`
`}`			

4.3 任务 2 图片文件的上传与显示

任务描述

一个图文并茂的网站，图片的上传与显示是最常用的功能之一。用户在上传图片时，有很多图片不符合要求，因此常需要对图片进行处理。本任务主要学习图片的裁剪、压缩和添加网站标记水印等常用的图片处理方式，同时掌握图片文件的两个存储方式。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）掌握 ASP.NET 中常见的两种图片文件的存储方式；

（2）掌握图片的基本处理方式；

（3）掌握图片在网页的显示方法。

4.3.1 GDI+ 中裁切和缩放图像

通常一个 Web 程序不会只把图片文件存放到指定的文件夹中，而是把图片存放在文件夹，同时把与图片相关的信息存放在数据库。下面的实战演练展示了把图片压缩后存放在数据表中，同时把图片的原图以文件的方式存放在文件夹中，并在数据表的相关字段中存放图片文件的路径。

一、实战演练

（1）在项目中添加图片上传页面，命名为“imagehandle.aspx”，页面设计如图 4-9 所示。

（2）双击“上传”按钮，添加按钮单击事件，如代码清单 4-8 所示。

图 4-9 图片上传页面

代码清单 4-8 “上传”按钮的单击事件代码

```
    protected void  BtnUpload_Click(object sender, EventArgs e)
    {
    SqlConnection  sqlconn = new SqlConnection(ConfigurationManager.Connection
    Strings["strconn");
        sqlconn.Open();
       string imagepath = Server.MapPath("~/images/");
       SqlCommand  sqlcomm=new SqlCommand("insert into addimagedata
    (imagename,imagedata,imagepath) values(@igname,@igdata,@imagepath)",sqlconn);
        if(FileUpload1.HasFile)
        {
              byte[] binaryimage;
              //更改上传文件的名字，文件名由时间组成
              string fileext = FileUpload1.FileName.Substring(FileUpload1.File
              Name.LastIndexOf('.'));
              string filename = DateTime.Now.Year.ToString() + DateTime.Now.Month.
              ToString() +
                DateTime.Now.Day.ToString() + DateTime.Now.Hour.ToString() +
    DateTime.Now.Minute.ToString() + DateTime.Now.Second.ToString()+fileext;
               imagepath = imagepath + filename;
              //把上传的图片内容导入一个 Bitmap 类中
              Bitmap bitm = new Bitmap(FileUpload1.FileContent);
              //判断文件大小，只要高、宽中任一个大于 200，对图片按原比率进行压缩
              if (bitm.Height > 200 || bitm.Width > 200)
              {
              //根据原图片的宽、高比率设置新图片的宽和高
                 double rate =(double) bitm.Width /(double) bitm.Height;
                 int newwidth = 200;
                 int newheigh = Convert.ToInt32( 200 * rate);
              //根据指定图片新的宽和高压缩图片
                 bitm = KiResizeImage(bitm, newwidth, newheigh, 1);
                 System.IO.MemoryStream ms = new System.IO.MemoryStream();
              //根据上传的图片类型选择压缩后的图片生成类型
                 switch (fileext)
                 {
                    case ".jpg":
                       bitm.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg);
                       break;
                    case ".gif":
                       bitm.Save(ms, System.Drawing.Imaging.ImageFormat.Gif);
break;
                 }
              //将图片数据写入缓存
                 binaryimage = ms.GetBuffer();
              //把原图以文件的形式存放指定的文件夹，地址写入数据库
                 FileUpload1.SaveAs(imagepath);
                sqlcomm.Parameters.Add("@imagepath", SqlDbType.NVarChar).Value =
                "images/" +
                filename;
                }
               else
```

```
            {  //如果图片比较小，直接将原图放入，图片的地址字段设置为空
               binaryimage = FileUpload1.FileBytes;
              sqlcomm.Parameters.Add("@imagepath", SqlDbType.NVarChar).Value =
              DBNull.Value;
            }
             //将图片数据和文件名写入数据库
              sqlcomm.Parameters.Add("@igname", SqlDbType.VarChar).Value =
             FileUpload1.FileName;
           sqlcomm.Parameters.Add("@igdata", SqlDbType.Image).Value = binaryimage;
             try
             {
                 sqlcomm.ExecuteNonQuery();
                Label1.Text="上传成功";
             }
             catch (Exception ex)
             {
                 Label1.Text = "上传不成功，请重新上传";
             }
             finally { sqlcomm.Dispose(); sqlconn.Close(); }
    }
}
             //图片压缩方法
public static Bitmap KiResizeImage(Bitmap bmp, int newW, int newH, int Mode)
{
       Bitmap b = new Bitmap(newW, newH);
       Graphics g = Graphics.FromImage(b);
           //插值算法的质量
       g.InterpolationMode =System.Drawing.Drawing2D.InterpolationMode.
       HighQualityBicubic;
       g.DrawImage(bmp, new Rectangle(0, 0, newW, newH), new Rectangle(0, 0,
       bmp.Width,
bmp.Height), GraphicsUnit.Pixel);
       g.Dispose();
       return b;
}
```

（3）单击工具栏中的“运行”按钮，进行程序调试。测试能否成功实现把图片文件分别以数据方式和文件方式上传到数据库和指定目录。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-4 所示。

表 4-4　演练完成情况评价表

任　务　号	4-4	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
图片压缩			
图片上传			

三、知识点

在图片处理中图片的缩放和剪裁是常见的两种处理方式，主要使用 Graphics. DrawImage（Image image,Rectangle destRect,Rectangle srcRect,GraphicsUnit srcUnit）方法来实现。缩放的原理是把整个原始图都往目标区域里塞，剪裁只是把原始区域上等宽等高的那个区域塞到目标区域里。

决定图片缩放时的质量与缩放图像时使用的算法有关，可以通过设置 Graphics.

InterpolationMode 属性值来完成，该属性值是 System.Drawing.Drawing2D 命名空间的 InterpolationMode 枚举值。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 3

拓展任务号	4-3	任务名称	实现图片的剪裁
计划用时	30 分钟	任务性质	课内
任务描述与目标			
对指定的图片进行剪裁，进一步巩固图片处理知识			
主要操作步骤提示			

1．右击项目，添加一个图片处理类 imagedeal.cs;

2．在类文件中，添加图片的剪裁的方法，参考代码如下：

```
public static Bitmap KiCut(Bitmap b, int StartX, int StartY, int iWidth, int iHeight)
{
         if (b == null)
          {
                 return null;
           }
      int w = b.Width;
      int h = b.Height;
      if (StartX >= w || StartY >= h)  {return null;  }
  if (StartX + iWidth > w){  iWidth = w - StartX;            }
     if (StartY + iHeight > h)  {  iHeight = h - StartY;            }
      try
     {
       Bitmap bmpOut = new Bitmap(iWidth, iHeight, PixelFormat.Format24bppRgb);
       Graphics g = Graphics.FromImage(bmpOut);
       g.DrawImage(b, new Rectangle(0, 0, iWidth, iHeight), new Rectangle(StartX, StartY, iWidth,
iHeight), GraphicsUnit.Pixel);
                 g.Dispose();
                 return bmpOut;
         }
       catch
      {  return null; }
}
```

4.3.2 图片的显示

一、实战演练

（1）在项目中添加显示图片页面，命名为“imageAccess.aspx”，页面设计如图 4-10 所示。设计页面的主要代码如代码清单 4-9 所示。

图 4-10 显示图片页面

代码清单 4-9 显示图片设计代码

```
<asp:Button ID="Button1" runat="server" onclick="Button1_Click" Text="显示
```

```
图片" />
    <asp:DropDownList ID="DropDownList1" runat="server" AutoPostBack=" true"
    Height="16px" onselectedindexchanged="DropDownList1_SelectedIndex Changed" >
          </asp:DropDownList> <br />
    <a id="a1" runat="server"> <asp:Image ID="Image1" runat="server"  /></a>
```

（2）添加以数据方式保存的显示图片页面，文件名为“showimages.aspx”，双击页面的空白处，添加 Page_Load 事件，如代码清单 4-10 所示。

代码清单 4-10　显示图片数据代码

```
    protected void Page_Load(object sender, EventArgs e)
      {
          string ID = Request.QueryString["ID"];
          if (ID == null) return;
         SqlConnection sconn = new SqlConnection(ConfigurationManager.Connection
Strings["strconn"]);
          sconn.Open();
          SqlCommand cmd = new SqlCommand("select imagedata from addimagedata
where imageID=" + ID, sconn);
          if (!(Convert.IsDBNull(cmd.ExecuteScalar())))
            {
               byte[] buffer = (byte[])cmd.ExecuteScalar();
               sconn.Close();
               Response.BinaryWrite(buffer);
               Response.End();
            }
        }
```

（3）选择 imageAccess.aspx.cs 文件，在“源”视图中添加 showimage()方法，用于显示以文件方式存放的图片文件，如代码清单 4-11 所示。

代码清单 4-11　showimage()方法代码

```
    void showimage(int id)
    {
    a1.HRef = null;
           Image1.AlternateText = null;
       try
       {
         SqlConnection sqlconn = new SqlConnection(ConfigurationManager.Connection
         Strings["strconn"]);
         sqlconn.Open();
         SqlCommand sqlcomm = new SqlCommand("select imagepath from addimagedata
        where imageID= @id", sqlconn);
         sqlcomm.Parameters.Add("@id", SqlDbType.Int).Value = DropDownList1.
         SelectedItem.ToString();
          SqlDataReader sqlsr = sqlcomm.ExecuteReader();
    //假如图片存在，在显示同时替代文本
          if (sqlsr.Read() && sqlsr.GetValue(0) != DBNull.Value)
           {
              string imgpath = sqlsr.GetValue(0).ToString();
              a1.HRef = imgpath;
              Image1.AlternateText = "单击看大图";
           }
          sqlsr.Close();
          sqlconn.Close();
```

```
        }
          catch { }
    }
```

（4）选择“imageAccess.aspx”，双击“显示图片”按钮，添加按钮单击事件，把图片的 ID 显示在下拉列表控件中，如代码清单 4-12 所示。

代码清单 4-12 “显示图片”按钮事件代码

```
    protected void Button1_Click(object sender, EventArgs e)
     {
    try
    {
    SqlConnection sqlconn = new SqlConnection(ConfigurationManager.Connection
Strings["strconn"]);
    sqlconn.Open();
    SqlCommand sqlcomm=new SqlCommand("select imageID,imagepath from addimagedata
",sqlconn);
       SqlDataReader sqlsr = sqlcomm.ExecuteReader();
       DropDownList1.Items.Clear();
       while (sqlsr.Read())
           {
             DropDownList1.Items.Add(sqlsr.GetValue(0).ToString());
           }
         sqlsr.Close();    sqlconn.Close();
    //显示在下拉列表中选中 ID 的图片
        Image1.ImageUrl = @"showImage1.aspx?ID=" + DropDownList1.SelectedItem.
ToString();
       showimage(Convert.ToInt32(DropDownList1.SelectedItem));
       }
        catch { }
    }
```

（5）选择 DropDownList1 控件，在“事件”面板中双击 SelectedIndexChanged 事件，并给事件添加如代码清单 4-13 所示的代码。

代码清单 4-13 DropDownList1 控件事件代码

```
    protected void DropDownList1_SelectedIndexChanged(object sender, EventArgs e)
    {
    Image1.ImageUrl = @"showImage1.aspx?ID=" + DropDownList1.SelectedItem.
ToString();
         showimage(Convert.ToInt32( DropDownList1.SelectedItem.ToString()));
    }
```

（6）运行程序，单击“显示图片”按钮，运行界面如图 4-11 所示。

（7）如果图片是经过压缩后存放的，单击图片可以查看大图，如图 4-12 所示；如果选择的图片不存在大图，则不能通过单击图片查看。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-5 所示。

图 4-11　单击“显示图片”按钮后的运行界面

图 4-12　单击查看大图界面

表 4-5　演练完成情况评价表

任　务　号	4-5	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
显示数据方式存放的图片			
显示文件方式存放的图片			

三、知识点

图片是目前常见的一种数据类型，是 Web 项目开发中经常会涉及的问题。一般来说图片的存储主要有两种方式：一种是把图片放在项目指定的目录中，把图片的地址放在数据库中；另一种是把图片数据直接存放在数据库中。这两种存放方式也各有利弊，主要还是根据图片大小、量的多少及项目的管理情况而定。

由于图片的数据量比较大，采用直接数据存储方式会造成整个数据库访问速度变缓，但是在进行信息记录管理时比较方便。因此一般在图片与其他信息关系比较密切且图片相对比较小时采用这种方式存储，如会员照片。如果图片比较大、同其他数据信息关系松散、多个数据记录使用同一个图片等这些情况，那么主要采用文件方式存储比较好。其缺点是管理比较麻烦，当数据库里其他相关信息被删除时，同时要根据情况在代码中实现图片的同步删除。

注意：文件方式存放图片上一般是通过 FileUpload1.SaveAs()方法与图片的其他信息分步上传的，如果其他信息在写入数据库时失败，就不能执行 SaveAs()方法。

一个 Web 程序通常情况是把图片存放在文件夹，同时把与图片相关的信息存放在数据库。以数据方式存放的图片要在 Web 页面中显示则比较麻烦，一般采用把二进制图片数据通过 Response.BinaryWrite()方法输出到 Web 页面中，再设置 ImageUrl 指向显示图片的 Web 页面，代码如下：

```
    <asp:Image ID="Image1" runat="server" ImageUrl='<%#"showImage.aspx?ID="+
Eval("imageID ") %>' />
```

以文件方式存放的图片比较简单，只要将图片控件 Url 属性指向图片的地址就可以了。

4.4　任务 3　文字处理与第三方控件的使用

文字是网站信息最重要的组成元素，网站的信息是由浏览器利用网站中的在线编辑器上传各类信息，这才使得网站的内容能及时更新，网站才有了生命力。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）掌握简易文本编辑器开发原理；
（2）了解目前常见的第三方控件的使用方式；
（3）掌握目前常见的两个图文编辑器 FreeTextBox 和 CuteEditor 的使用。

4.4.1 实现简易文本编辑器

一、实战演练

（1）在项目中添加 Web 页面，命名为“infohandle.aspx”，在页面中添加四个 HTML 控件标记，一个<iframe>标记，一个 Button 服务器控件，一个隐藏的服务器控件，一个 Lable 服务器控件，在<iframe>标记上方添加四个 HTML 按钮，设计如图 4-13 所示。

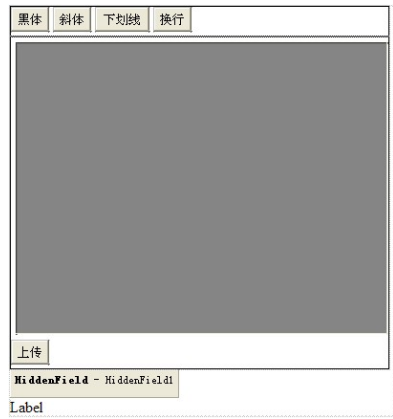

图 4-13 简易文本编辑器界面设计

（2）切换到“源”视图，为相应的各个控件添加客户端脚本代码，如代码清单 4-14 所示。

代码清单 4-14 简易文本编辑器脚本代码

```
<script language="javascript">
 //设置选中文字加粗
 function addBold()
 {
  testedit.focus();
  var sel=testedit.document.selection.createRange();
  sel.execCommand("Bold");
 }
 //设置选中文字为斜体
 function addItalic()
 {
   testedit.focus();
    var sel=testedit.document.selection.createRange();
   insertHTML("<i>"+sel.text+"</i>");
 }
  //设置选中文字下画线
  function addUnderL()
 {
   testedit.focus();
    var sel=testedit.document.selection.createRange();
   insertHTML("<U>"+sel.text+"</U>");
 }
 //在鼠标指针处添加换行
  function addEnter()
 {
   testedit.focus();
    var sel=testedit.document.selection.createRange();
   insertHTML("<br/>"+sel.text);
 }
 //把添加标记的文本插入文本选中处
 function insertHTML(html)
 {
   if(testedit.document.selection.type.toLowerCase()=="none")
      testedit.document.selection.clear();
    testedit.document.selection.createRange().pasteHTML(html);
```

```
    }
    //返回设置完成的文本
     function   SetIframe()
          {
          var
content=document.getElementById( "testedit").contentWindow.document.body.inne
rHTML;
          alert(content);
           document.getElementById( "HiddenField1").value=content;
          }
   </script>
```

（3）在各个相应的控件中添加相关的客户端事件代码，为< iframe>标记添加 onload 事件，在事件中设置 iframe 为可编辑模式，并添加客户端事件，如代码清单 4-15 所示。

代码清单 4-15　客户端事件代码

```
   <body >
      <form id="form1" runat="server" onsubmit="SetIframe()">
      <div style="border-style: solid; width:395px;  border-width: 1px">
      <input id="Button1" type="button" value="黑体" onclick="addBold()" />
      <input id="Button3" type="button" value="斜体" onclick="addItalic()" />
   <input id="Button2" type="button" value="下画线" onclick="addUnderL()"  />
      <input  id="Button5"  type="button"  value=" 换 行 "  onclick="addEnter()"
/></div>
   <div style="border-style: solid;height:346px; width:389px;  border-width:
1px">
   //将 iframe 设置为可编辑
   <iframe id="testedit"
         style= "height:350px; width: 386px; background-color:#FFFFFF;"
         marginheight="0" onload="testedit.document.designMode='on'">
   </iframe>
   <asp:Button ID="submitbtn" runat="server" Text="上传" onclick="submitbtn_Click"
/>
      </div><asp:HiddenField   ID= "HiddenField1"   runat= "server"   />
      <asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
      </form>
   </body>
```

（4）切换到“设计”视图，双击“上传”按钮，添加事件，如代码清单 4-16 所示。

代码清单 4-16　“上传”按钮事件代码

```
   protected void submitbtn_Click(object sender, EventArgs e)
    {
      string str = HiddenField1.Value;
   //实现显示< iframe>中编辑的文字，也可以转换为上传到数据库
      Label1.Text = str;
    }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-6 所示。

表 4-6　演练完成情况评价表

任务号	4-6	任务名称	简易文本编辑器
完成情况		主要问题	未完成原因

三、知识点

ASP.NET 所提供的文本控件在上传文本信息时，不能实现分段、空行等功能。在实际的应用中，采取的是曲线的方式，把<iframe>标记通过属性设置成为可编辑状态，通过在文本中插入 HTML 的格式标记来实现。

由于简易文本编辑器中上传的文本信息包含<html>标记的内容，在客户端中会检测到有潜在危险的 Request.Form 值，因此需要设置如下 Page 指令：

```
<%@ Page language="c#"……ValidateRequest="false"%>
```

4.4.2 FreeTextBox 上传组件的应用

FreeTextBox 是一款免费的 ASP.NET 网页编辑器，官方默认为英文版，可以设置文字样式、在线排版、图片上传等。

一、实战演练

（1）在项目中添加一个文件夹 freeEdit，在此文件夹下添加 Bin 文件夹，将 FreeTextBox.dll 文件到复制到 Bin 目录中；并在 freeEdit 下建立 images 文件夹，作为上传图片的图片库。

（2）在文件夹中添加一个 Web 页面，命名为“freeEditBox.aspx”。

（3）选择工具箱中的“常规”选项卡，在菜单栏中执行“工具”→“选择工具箱”命令，打开如图 4-14 所示的“选择工具箱项”对话框。

（4）在对话框中选择“.NET Framework 组件”选项卡，单击对话框右下角的“浏览”按钮，在弹出的“打开文件”对话框中找到应用程序项目中的“freeEdit”下的“Bin”文件夹，选择 FreeTextBox.dll 后，可以在“选择工具箱项”对话框中看到添加的 FreeTextBox 控件。同时在如图 4-15 所示的“常规”选项卡中可以看到新添加的 FreeTextBox 控件和 ImageGallery 组件。

图 4-14 “选择工具箱项”对话框

图 4-15 “常规”选项卡

（5）打开 freeEditBox.aspx 文件，在“设计”视图中把 FreeTextBox 控件拖入页面，并拖入一个 Label 控件和一个 Button 控件。这时页面上只能看到 FreeTextBox 控件的一个占位符。运行调试，可以看到如图 4-16 所示的页面显示效果。

（6）在 FreeTextBox 控件的上方是对文本框内的文本进行格式设置的按钮或选择框，通过单击这些按钮可以对选中的文本进行设置。在 FreeTextBox 控件的下方有两个选择框，单击“Design”看到的是设计效果，而单击“HTML”可以看到相应的 HTML 代码。

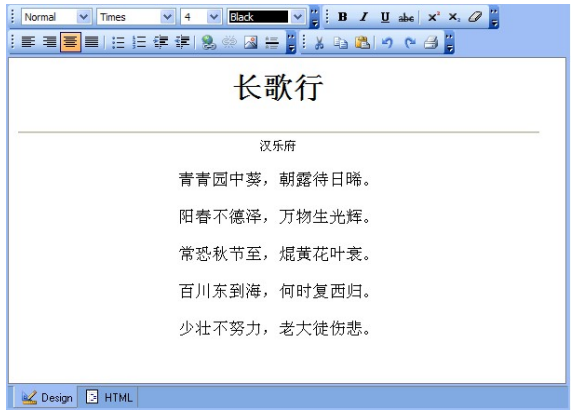

图 4-16　运行 FreeTextBox 控件后的显示效果

（7）默认设置的 FreeTextBox 控件不能实现图文同时上传的功能，如果希望能使用更多功能则需要设置 FreeTextBox 控件的 Toolbarlayout 属性。完成属性设置后的 FreeTextBox 控件代码如代码清单 4-17 所示。

代码清单 4-17　FreeTextBox 控件代码

```
    <FTB:FreeTextBox ID="FreeTextBox1" Focus="true" ButtonDownImage="true" Width="
600px" ImageGalleryPath="~/freeEdit/images" ImageGalleryUrl="imagegallery. aspx?rif=
{0}&cif={0}"     Toolbarlayout= "ParagraphMenu,FontFacesMenu,FontSizesMenu, FontFore
ColorsMenu,FontForeColorPicker,FontBackColorsMenu,FontBackColorPicker;
           |CreateLink,Unlink,Outdent,Indent;
    |Bold,Italic,Underline,Strikethrough,Superscript,Subscript,RemoveFormat;
    |JustifyLeft,JustifyRight,JustifyCenter,JustifyFull;BulletedList,Numbered
List,Indent,InsertImage;
           |InsertRule,InsertDate,InsertTime;
           |Cut,Copy,Paste,Delete;Undo,Redo,Print;
           |SymbolsMenu,InsertHtmlMenu;
           |InsertTable,EditTable;
           |InsertImageFromGallery,Preview,SelectAll,WordClean,NetSpell;"
runat="server">
          </FTB:FreeTextBox>
```

（8）设置后的 FreeTextBox 控件，功能更加强大，不仅可以对文本进行格式设置，同时还能实现图文同步上传。要实现这个功能，需要再添加一个图片处理文件 imagegallery.aspx。

（9）打开 imagegallery.aspx 文件，把“常规”选项卡中的 ImageGallery 组件拖入页面中，在“设计”视图中会出现“创建控件出错”的字样。切换到“源”视图修改 HTML 代码，并对 ImageGallery 组件进行属性设置。设置完成的 imagegallery.aspx 文件代码如代码清单 4-18 所示。

代码清单 4-18　imagegallery.aspx 文件代码

```
    <%@ Page Language="C#" %>
    <%@ Register Assembly="FreeTextBox" Namespace="FreeTextBoxControls" Tag
Prefix="FTB" %>
    <%--注意下面语句要作注释--%>
    <%--<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/ TR/xhtml1/DTD/xhtml1-transitional.dtd">--%>

    <html xmlns="http://www.w3.org/1999/xhtml">
    <head runat="server">
       <title>无标题页</title>
    </head>
```

```
    <body>
        <%--from 标记一定要设置 enctype 属性--%>
        <form id="form1" runat="server" enctype="multipart/form-data">
        <div>
            <FTB:ImageGallery ID="ImageGallery1" runat="server" AllowImageDelete="True" AllowImage
Upload="True" AllowDirectoryDelete="True" AllowDirectoryCreate="True">
            </FTB:ImageGallery>
        </div>
        </form>
    </body>
    </html>
```

（10）打开 freeEditBox.aspx 文件，设置 FreeTextBox 控件属性如下：

```
    ImageGalleryPath= "~/freeEdit/images"
    ImageGalleryUrl="imagegallery.aspx?rif={0}&cif={0}"
```

（11）运行调试，可以看到如图 4-17 所示的页面显示效果。

（12）单击图 4-17 中画圈处的“插入图片”按钮，弹出如图 4-18 所示的图片插入页面。通过单击页面下方的“浏览”按钮可以选择图片，单击“Upload”按钮上传到指定的图片目录，并在页面中显示。选择相应的图片后，在页面的右边进行相关的设置后单击“Insert”按钮，可以在 FreeTextBox 控件中看到拖入图片的效果。

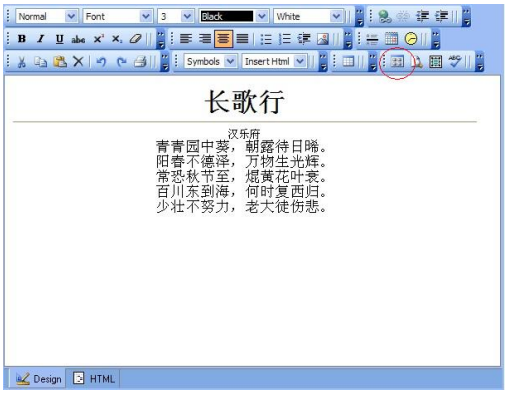

图 4-17　对 Toolbarlayout 属性设置后的页面显示效果

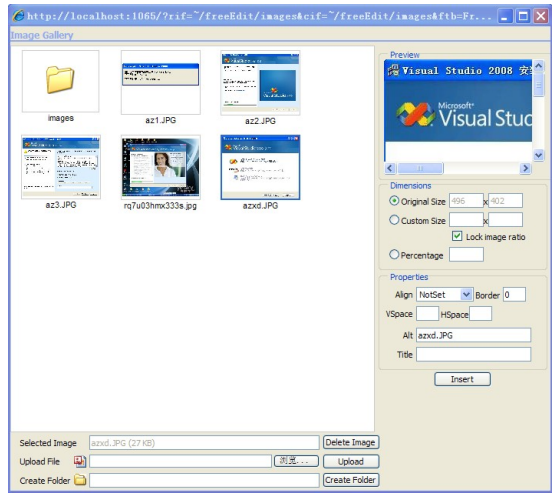

图 4-18　图片插入页面

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-7 所示。

表 4-7　演练完成情况评价表

任　务　号	4-7	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
FreeTextBox 控件的设置与新部署			
FreeTextBox 控件的应用			
ImageGallery 组件的应用			

三、知识点

要使用 FreeTextBox 控件，一般先把控件添加到工具栏，直接把控件拖拽到 Web 窗体中，在 HTML 代码页页头会自动添加<%@ Register TagPrefix="ftb" Namespace= "FreeTextBoxControls" Assembly="FreeTextBox"%> 指令和引入控件标签 <FTB:FreeTextBox id= "FreeTextBox1" runat="server" Width="500px" Height="400px" />。

由于 FreeTextBox 控件实现时客户端中会检测到有潜在危险的 Request.Form 值，因此需要设置 Page 指令为：

```
<%@ Page language="c#" ... ValidateRequest="false"%>
```

FreeTextBox3.0 以上的版本均支持内部模式，即图片资源和 JavaScript 都集成在 dll 中，也可以使用外部模式，如果使用外部模式则需要把 FreeTextBox 的支持文件及文件夹复制到应用程序中，不过这要求对 FreeTextBox 非常了解，如何设置读者可以参考其他文献知识。FreeTextBox 控件最常用的属性设置是 toolbarlayout，该属性主要设置编辑框上方的可用功能按钮。Toolbarlayout 的完整属性值为：

Toolbarlayout="ParagraphMenu,FontFacesMenu,FontSizesMenu,FontForeColorsMenu,
FontForeColorPicker,FontBackColorsMenu,FontBackColorPicker;
|Bold,Italic,Underline,Strikethrough,Superscript,Subscript,RemoveFormat;
|JustifyLeft,JustifyRight,JustifyCenter,JustifyFull;BulletedList,NumberedList,Indent,Outdent;
|CreateLink,Unlink,InsertImage;
|Cut,Copy,Paste,Delete;Undo,Redo,Print,Save;
|SymbolsMenu,StylesMenu,InsertHtmlMenu;
|InsertRule,InsertDate,InsertTime;
|InsertTable,EditTable;InsertTableRowAfter,InsertTableRowBefore,DeleteTableRow;
InsertTableColumnAfter,InsertTableColumnBefore,DeleteTableColumn;
|InsertForm,InsertTextBox,InsertTextArea,InsertRadioButton,InsertCheckBox,InsertDropDown List,InsertButton;
|InsertDiv,EditStyle,InsertImageFromGallery,Preview,SelectAll,WordClean,NetSpell"。

如果在 FreeTextBox 控件的工具栏上添加 ImageGallery 按钮，则 FreeTextBox 控件也会支持图片上传。图片上传是通过 ImageGallery 组件实现的，这个控件应放在同一个目录的另一个文件中，如 ImageGallery.aspx。在 FreeTextBox 控件中设置 ImageGalleryPath 和 ImageGalleryUrl 属性的设置格式如下：

```
ImageGalleryPath = "~/image/upload"     //设置上传图片的默认路径
//设置图片处理文件 ImageGallery.aspx 的目录，只能用相对目录，不可以用“~”
ImageGalleryUrl = " ImageGallery.aspx?rif={0}&cif={0}"
```

ImageGallery 组件常用格式为：

```
<FTB:ImageGallery id="ImageGallery1"JavaScriptLocation="InternalResource"
UtilityImages Location="InternalResource"SupportFolder="~/aspnet_client/Free
TextBox/"AllowImageDelete=true AllowImageUpload=true AllowDirectoryCreate=
false AllowDirectoryDelete=false runat="Server"/>
```

FreeTextBox 控件上传组件包含 FreeTextBoxControls、FreeTextBoxControls.Design 和 FreeText BoxControls.Common 三个命名空间。

注意：在建立图片上传文件时一定要给<!DOCTYPE>这句添加注释，也可以直接删除。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 4

拓展任务号	4-4	任务名称	FreeTextBox 上传组件使用练习
计划用时	30 分钟	任务性质	课内
任务描述与目标			
Toolbarlayout 是设置 FreeTextBox 上传组件功能的属性，通过对 Toolbarlayout 设置的练习，熟练掌握 FreeTextBox 控件的使用			
主要操作步骤提示			
1．打开“freeEditBox.aspx”文件； 2．转到“源”视图； 3．修改 Toolbarlayout 的属性值，观察修改后的变化； 4．添加、删除 Toolbarlayout 属性值中的“|”符号，观察“|”符号的作用； 5．FreeTextBox 上传组件提供外置功能，外置功能提供更丰富的样式，可以练习使用			

4.4.3 用 CuteEditor 组件实现数据与文件的同步上传

CuteEditor 是一款功能非常强大，支持图片上传、文件下载和 Word 类似的文字编辑器。对于新闻发布系统和博客之类的系统，是非常方便的。

一、实战演练

（1）打开项目，把 CuteEditor 文件包中 Bin 文件夹的内容复制到项目根目录中。

（2）在项目根文件夹中，添加文件夹 CuteEdits。将 CuteEditor 文件包中的 example.css 文件和 CuteSoft_Client 文件夹复制到文件夹 CuteEdits 中。

（3）打开 Web.config 文件，在文件中添加如下代码：

```
<appSettings>
    <add key="DictionaryFolder" value="bin"/>
</appSettings>
```

（4）在文件夹 CuteEdits 下建立 Uploads 和 Templates 两个文件夹，分别用于存放上传的图片及附件。

（5）打开 CuteSoft_Client\CuteEditor\Configuration\Security 文件夹中的 Admin.config 文件，修改上传文件存放地址，修改代码如下：

```
<security name="ImageGalleryPath">~/CuteEdits/uploads</security>
<security name="MediaGalleryPath">~/CuteEdits/uploads</security>
<security name="FlashGalleryPath">~/CuteEdits/uploads</security>
<security name="TemplateGalleryPath">~/CuteEdits/templates</security>
<security name="FilesGalleryPath">~/CuteEdits/uploads</security>
```

（6）执行“工具”→“选择工具箱”命令，将 CuteEditor.dll 添加到工具箱中。

（7）在文件夹 CuteEdits 中添加 CuteEDT.aspx 文件，把工具箱中的 Editor 控件拖入页面中，在页面中只能看到一个占位符。在 Editor 控件下方添加一个 Label 控件和一个 Button 控件。双

击 Button 控件，在事件中添加代码如下：

```
Label1.Text = Editor1.Text;
```

（8）运行文件 CuteEDT.aspx 浏览效果，效果如图 4-19 所示。输入文本后可以单击“Button”按钮查看 Editor 控件中编辑的文本内容在页面中的显示效果。

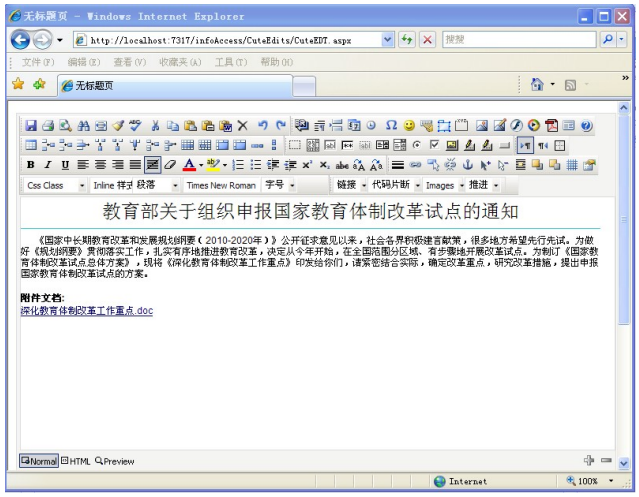

图 4-19　Editor 控件页面的运行效果

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 4-8 所示。

表 4-8　演练完成情况评价表

任　务　号	4-8	任 务 名 称	开发前准备
任务子项	完成情况	主要问题	未完成原因
CuteEditor 组件设置与部署			
CuteEditor 控件的应用			

三、知识点

CuteEditor 是一款很好的 Web 在线编辑器，功能非常强大。在下载文件包的 Bin 文件夹中包含以下四个文件：

- ✧ 主文件 CuteEditor.dll；
- ✧ 增加的 EditorImage 功能文件 CuteEditor.ImageEditor.dll；
- ✧ 解密文件 CuteEditor.lic；
- ✧ 拼写检查功能文件 NetSpell.SpellChecker.dll。

注意：以.lic 为扩展名的文件是保存为纯文本文件的格式。将 CuteEditor 6.0 中 Bin 文件夹里的内容都复制到站点根目录下的 Bin 文件夹中。

CuteSoft_Client 文件夹包含 CuteEditor 文件和 example.css 文件，这两个文件应该复制到与使用 CuteEditor 控件的 Web 文件相同的目录中。

一般来说，使用 CuteEditor 控件的文件一般不在项目的根目录下，需要对 CuteEditor 控件的 FilesPath 和 EditorWysiwygModecss 属性设置所对应的目录，如：

```
FilesPath="~/admin/CuteSoft_Client/CuteEditor/"
EditorWysiwygModecss="~/admin/CuteSoft_Client/example.cs"
```

CuteEditor 控件支持各类文件的上传，不同文件上传的路径设置通过 CuteSoft_Client\CuteEditor\Configuration\Security 文件夹中的 Admin.config、Default.config、Guest.config 来控制上传图片和文件的问题。

目前 CuteEditor 控件的版本很多，不同的版本配置方式也许有些不同，但原理基本相同，在实际应用中多练习可以熟练掌握。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 5

拓展任务号	4-5	任务名称	CuteEditor 上传组件的练习
计划用时	30 分钟	任务性质	课内

任务描述与目标

CuteEditor 控件是一款很好的 Web 在线编辑器，特别是支持进行多种分类文件的上传及管理。CuteEditor 文件包中提供了帮助文件及参数设置文件，通过对 CuteEditor 上传参数的修改熟练掌握 CuteEditor 控件的使用原理

主要操作步骤提示

1. 打开 CuteSoft_Client\CuteEditor\Help\ default.aspx 文件，转到“设计”视图，可以看到这是一个帮助文件，在文件中对 CuteEditor 控件的各功能按钮的作用进行了说明；

2．打开 CuteSoft_Client\CuteEditor\Configuration\Security 文件夹；

3．打开 Default.config 文件，其中有三个<security>节分别控制三种上传方式的文件类型。修改下面三个节后运行 CuteEDT.aspx 文件，观察修改后组件功能的变化。

注意：在修改前先要对 Default.config 文件进行备份。

```
<security name="ImageFilters">
        <item>.jpg</item>
        <item>.jpeg</item>
         <item>.gif</item>
<item>.png</item>
   </security>
   <security name="MediaFilters">
        <item>.avi</item>
        <item>.mpg</item>
        <item>.mpeg</item>
        <item>.mp3</item>
   </security>
   <security name="DocumentFilters">
        <item>.txt</item>
        <item>.doc</item>
        <item>.pdf</item>
        <item>.zip</item>
        <item>.rar</item>
        <item>.avi</item>
        <item>.mpg</item>
        <item>.mpeg</item>
        <item>.swf</item>
   </security>
```

拓展任务跟踪卡

任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

项目号			项目名	
项目完成方式			□小组协作　　□个人独立	
项目完成情况（60%）	界面设计（2%）	自我评价		
		小组评价		
		个人评价		
	代码编写（30%）	自我评价		
		小组评价		
		个人评价		
	功能实现（20%）	自我评价		
		小组评价		
		个人评价		
	说明文档（5%）	自我评价		
		小组评价		
		个人评价		
	数据库设计(3%)	自我评价		
		小组评价		
		个人评价		
拓展与创新（40%）	创新能力（30%）	自我评价		
		小组评价		
		个人评价		
	设计潜力（10%）	自我评价		
		小组评价		
		个人评价		
主要存在问题				

课外思考题

1．目前需要开发管理用户数据的应用程序，该应用程序需要接受用户输入的信息，将其组织成一条基本信息，然后将其存储在应用程序目录子目录下的一个文件中。已经制定了一个文件命名约定，现在请完成代码将用户的数据写入文件。

2．接第 1 题，现在要对之前创建的用户信息数据应用程序进行扩展，以允许用户搜索数据文件。已经确定必须使用 FileStream 来连接文件进行读数据，但是需要允许用户搜索文件并根据搜索结果从文件返回数据段，并向页面中显示其数据的应用程序。请编写代码实现此功能。

项目五

企 业 网 站

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 掌握 ASP.NET 网站布局，网站常见文件类型；
- ✧ 掌握 ASP.NET 母版页、主题、皮肤技术的使用；
- ✧ 掌握网站的 Web.config 文件的配置；
- ✧ 掌握 Application、Session、Cookie 等常用内置对象的使用；
- ✧ 掌握利用 ADO.NET 的非连接对象模型对数据的操作方法；
- ✧ 掌握数据绑定技术及常用数据控件的使用；
- ✧ 了解 Web Service 技术，掌握 Web Service 的定义与调用。

技能目标

通过对本项目的学习，应该具备下面的能力：

- ✧ 能根据项目需求，进行网站的整体规划；
- ✧ 能利用母版页等相关技术定制网站的模板，统一网站风格；
- ✧ 能进行网站的安全等相关配置；
- ✧ 能使用 ADO.NET 的非连接对象进行数据操作；
- ✧ 能利用 Web Service 技术整合其他项目功能，提高复用性和耦合性。

教学建议

本项目计划总学时为 22 学时。

- ✧ 情境介绍：2 学时；
- ✧ 任务 1：4 学时；
- ✧ 任务 2：2 学时；
- ✧ 任务 3：6 学时；
- ✧ 任务 4：4 学时；
- ✧ 任务 5：2 学时。
- ✧ 任务 6：2 学时。

企事业网站的功能大同小异，在项目开始时，教师可以让学生浏览自己比较熟悉行业的企事业网站，根据自己的兴趣选取这些企业网站作为蓝本，分析这些网站的主要功能，借鉴教材的内容进行开发。这样开展教学比纯粹使用教材中的项目案例更能激发学生的学习积极性。

在整个教学过程中，有很多功能是要求学生利用课余时间去实现、完善的。在项目结束后通过项目展示环节，进行项目的评定。

5.1 情境介绍

对于大多数中小企业来说，在互联网上建立网站已经不是要不要的问题，而是需要考虑如何做才能通过网站将企业在互联网中的形象树立起来。大多数人现在都开始通过互联网寻找产品、合作伙伴，企业网站已经成为市场竞争的最前沿阵地。一个企业的网站可以最直接反映企业的管理水平、市场宣传能力、以至于企业的核心实力。网站制作的不好、宣传不到位，直接影响企业市场竞争力。通过这个项目的开发，使得学生对于开发企业门户的网站有整体概念。

1. 需求分析

企业网站主要是向来访者提供企业产品服务信息，任何人都可通过网站获取这个企业的新闻、产品服务等资讯。而这些资讯信息是通过管理员登录进行管理的；因而网站分为前台和后台两部分，网站对应的角色也就分成了一般访问者和管理员两类。网站的前台和后台访问流程如图 5-1 和图 5-2 所示。

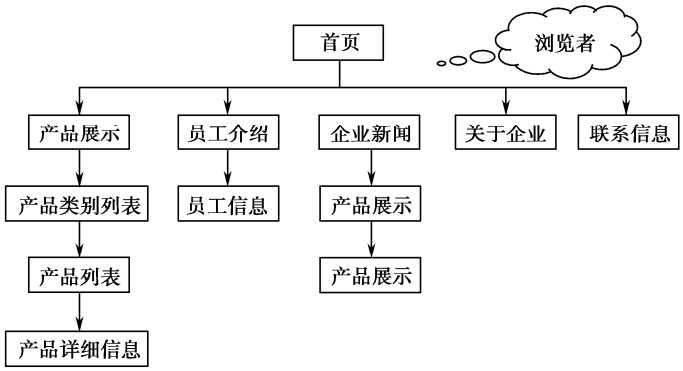

图 5-1 前台访问流程图

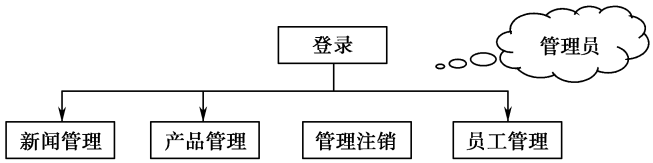

图 5-2 后台访问流程图

2. 功能设计

（1）产品服务。浏览者在相关栏目内，可以浏览企业提供的各类产品服务列表，可以查看某件产品服务的详细信息。可以按类别查看产品服务信息。

（2）新闻资讯。浏览者在相关栏目内，可以浏览企业相关的新闻资讯列表，可以查看某条新闻的详情。

（3）后台管理功能。在后台中主要进行新闻、产品资讯、企业员工信息的各类管理。

3. 数据库设计

1）表设计

根据企业网站项目的实际需求，创建数据库 Enterprise，建立一个如表 5-1 所示的管理员信息表（Admin），如表 5-2 所示的新闻文章表（Article），如表 5-3 所示的文章分类表（ArticleClass），如表 5-4 所示的客户意见反馈表（Feedback），如表 5-5 所示的通告表（Notice），

如表 5-6 所示的产品表（Product）和如表 5-7 所示的产品分类表（ProductClass）。

表 5-1　管理员信息表（Admin）

字 段 名	字 段 类 型	字 段 说 明	备　注
admin_id	int	管理员 ID	PK（自动增一）
username	nvarchar(50)	管理员名称	
password	nvarchar(50)	密码	
joindate	datetime	创建时间	
login_ip	nvarchar(50)	登录 IP	
login_date	datetime	登录时间	

表 5-2　新闻文章表（Article）

字 段 名	字 段 类 型	字 段 说 明	备　注
art_id	int	文章 ID	PK（自动增一）
art_title	nvarchar(50)	文章标题	
art_author	nvarchar(20)	作者	
art_from	nvarchar(50)	来源	
art_content	nvarchar(MAX)	正文	
art_description	nvarchar(100)	描述	
art_image	nvarchar(100)	图片	
art_date	datetime	发表时间	
art_click	int	单击数	
istop	bit	是否置顶	
iscommend	bit	是否推荐	
ac_id	int	类别 ID	外键（ArticleClass）

表 5-3　文章分类表（ArticleClass）

字 段 名	字 段 类 型	字 段 说 明	备　注
ac_id	int	文章类别 ID	PK（自动增一）
ac_name	nvarchar(50)	类别名称	
parent_id	int	父类 ID	
ac_order	int	类别顺序	

表 5-4　客户意见反馈表（Feedback）

字 段 名	字 段 类 型	字 段 说 明	备　注
fb_id	int	意见反馈 ID	PK（自动增一）
fb_title	nvarchar(50)	标题	
fb_name	nvarchar(20)	作者	
fb_phone	nvarchar(20)	电话	
fb_email	nvarchar(50)	E-mail	
fb_content	nvarchar(MAX)	意见正文	
fb_ip	nvarchar(50)	留言者 IP	
fb_date	datetime	留言时间	

表 5-5 通告表（Notice）

字 段 名	字 段 类 型	字 段 说 明	备 注
notice_id	int	ID	PK（自动增一）
notice_title	nvarchar(50)	标题	
notice_content	nvarchar(200)	正文	
notice_date	datetime	通告时间	

表 5-6 产品表（Product）

字 段 名	字 段 类 型	字 段 说 明	备 注
prod_id	int	文章 ID	PK（自动增一）
prod_name	nvarchar(50)	产品名称	
prod_number	nvarchar(50)	数量	
prod_price	money	单价	
prod_image	nvarchar(100)	图片	
prod_content	nvarchar(MAX)	描述	
prod_date	datetime	发布时间	
prod_click	int	单击数	
istop	bit	是否置顶	
iscommend	bit	是否推荐	
pc_id	int	类别 ID	外键（ProductClass）

表 5-7 产品分类表（ProductClass）

字 段 名	字 段 类 型	字 段 说 明	备 注
pc_id	int	产品类别 ID	PK（自动增一）
pc_name	nvarchar(50)	类别名称	
parent_id	int	父类 ID	
pc_order	int	类别顺序	

2）存储过程

为了提高程序的性能和数据库的访问速度，本项目中使用存储过程实现数据操作。

（1）打开 SQL Server 2008 企业管理器。在控制台目录中选择 Enterprise 数据库，并展开。

（2）右击“存储过程”节点，在弹出的快捷菜单中选择“添加新存储过程”选项，打开存储过程开发代码模式窗口。

（3）在该窗口中输入存储过程要实现的命令，如代码清单 5-1 所示。

代码清单 5-1 添加新管理员的存储过程代码

```
//用于添加新管理员的存储过程
Create PROCEDURE Sp_AdminInsert
(
   @username nvarchar(50),     //管理员的用户名
   @password nvarchar(50),     //密码
   @joindate datetime,         //创建时间
   @login_ip nvarchar(50),     //登录 IP
   @login_date datetime        //最近一次登录的时间
```

```
)
AS
BEGIN
    INSERT INTO
       Admin(username, password, joindate,
          login_ip, login_date)
    VALUES
        (@username,    @password, @joindate,
         @login_ip,    @login_date )
END
```

（4）单击“检查语法”按钮，如果提示“语法检查成功”，则表示代码清单 5-1 所示的代码没有语法问题。

（5）系统根据代码清单 5-1 中的存储过程名，生成一个名为 sp_AdminInsert 的存储过程。充分理解项目的开发需要，把常用的数据操作编写成存储过程，需要建立的参考存储及功能说明如下：

ResetOrderAc：在删除新闻或新闻分类后，重新设置每条新闻的序号

ResetOrderPc：在删除产品或产品分类后，重新设置每样产品的序号

Sp_AdminDeleteSingleItem：删除管理员 ID 为@admin_id 值的管理员信息

Sp_AdminInsert：添加一个新的管理员

Sp_AdminSelectItem：查询某一个管理员信息

Sp_AdminSelectList：查询所有管理员信息

Sp_AdminUpdateSingleItem：更新某个管理员信息

Sp_ArticleClassDeleteSingleItem：删除某个新闻类别

Sp_ArticleClassInsert：添加新闻类别

Sp_ArticleClassSelectItem：获取指定的某个新闻类别

Sp_ArticleClassSelectList：获取所有新闻类别

Sp_ArticleClassSelectListByParentID：获取某个新闻父类下的所有子类

Sp_ArticleClassUpdateSingleItem：更新某个新闻类别信息

Sp_ArticleDeleteSingleItem：删除某个新闻类别

Sp_ArticleInsert：添加一条新闻

Sp_ArticleSearchList：查找符合条件的新闻

Sp_ArticleSelectItem：获取某条新闻信息

Sp_ArticleSelectList：获取所有新闻信息

Sp_ArticleUpdateSingleClick：更新某条新闻的单击数

Sp-ArticleUpdateSingleItem：更新某条新闻的信息

Sp_FeedbackDeleteSingleItem：删除某条留言信息

Sp_FeedbackInsert：发表留言

Sp_FeedbackSelectList：获取留言列表信息

Sp_FeedbackUpdateSingleItem：更新留言信息

Sp_ProductClassDeleteSingleItem：删除某个产品类别

Sp_ProductClassGetTree：获取产品分类树

Sp_ProductClassInsert：添加一个新的产品类别

Sp_ProductClassSelectItem：查询某个产品类别信息

Sp_ProductClassSelectList：查询所有产品类别信息
Sp_ProductClassSelectListByParentID：查询某个产品父类下所有的子类
Sp_ProductClassUpdateSingleItem：更新一个产品类别信息
Sp_ProductDeleteSingleItem：删除一个产品类别信息
sprocProductInsert：添加一条新的产品
Sp_ProductSearchList：按条件查询产品信息
Sp_ProductSelectItem：查询一件商品信息
Sp_ProductSelectList：查询所有商品信息
sprocProductUpdateSingleClick：更新某件产品信息的单击数
Sp_ProductUpdateSingleItem：更新某件产品
Sp_SitePropertySelectItem：查询网站信息
Sp_SitePropertyUpdateItem：更新网站信息

5.2 任务 1 ASP.NET 网站结构

任务描述

在创建 ASP.NET 网站时，可以包含 ASP.NET 能够识别以进行处理的特定类型的文件。此外，还可以创建用于特殊用途（如用于存储源代码）的文件夹。本任务通过对企业网站布局学习 ASP.NET 网站文件布局方式。

解决方案

为完成本任务，要完成以下几个方面的工作：
1．能根据客户的需求按内容进行网站栏目结构设计；
2．能对 ASP.NET 的网站文件目录结构进行设计；
3．能基于提高可维护性和可扩展性的需求进行 Web.Config 的配置。

5.2.1 网站布局设计

网站结构设计从网站的整体架构、网站的版块、网站的整体色调、网站的素材选取等入手。网站首页是一个网站整体形象的体现，不同的行业对于网站首页的布局也是有不同要求的。优秀的布局设计，整个网站就显得非常漂亮，也就能够更好地吸引浏览者的纵深访问。

一、实战演练

（1）浏览 Internet 网上的各类企业网站，分析这些网站的首页整体布局。重点查看同类行业的网站。注意色彩的使用。

（2）分析企业的需求情况，形成网站栏目结构层次图。分析首页显示内容使用 Word 文字编辑软件确定各功能模块布局草图。

（3）根据行业访问者的惯性需求，确定网站的主色调，上网查找图片素材，并使用 Photoshop 等图片制作软件进行素材处理。

（4）确定网站设计的风格定位，使用图片制作软件设计出网站主要页面的布局。

（5）对设计完成的页面图片进行分割，以便在 Visual Studio 中进行页面布局设计。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-8 所示。

表 5-8　演练完成情况评价表

任　务　号	5-1	任 务 名 称	网站结构设计
任务子项	完成情况	主要问题	未完成原因
网站栏目结构图设计			
网站设计风格定位			

5.2.2　ASP.NET 文件类型

一、实战演练

（1）打开 VS 2012，创建网站项目，命名为 EnterPriseWeb。

（2）在“解决方案资源管理器”中，右键单击项目名称，在弹出的快捷菜单中选择“新建文件夹”选项。把新建的文件夹命名为 images，用于存放网站的各类图片素材。

（3）把建立好的数据库 Enterprise 存放在项目的 App_Data 目录中，在数据库管理器中重新附加数据库。

（4）在“解决方案资源管理器”中，右键单击项目名称，在弹出的快捷菜单中选择“添加新项”选项，在“添加新项”对话框中选择“全局应用程序类”，“名称”栏使用默认值 Global.asax，单击“确定”按钮。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-9 所示。

表 5-9　演练完成情况评价表

任　务　号	5-2	任 务 名 称	网站文件类型
任务子项	完成情况	主要问题	未完成原因
图像素材存放			
向 images 目录中导入图像素材			
创建全局应用程序类文件			
向 App_Data 目录中添加项目数据库文件			

三、知识点

在创建 ASP.NET 网站时，ASP.NET 自动创建了一些文件，同时也能根据需要创建识别进行处理的特定类型的文件。不同类型文件的功能及所在位置都是不同的。

1. 默认页

通过为 Web 应用程序建立默认页，使用户更容易定位到您的站点。默认页是在用户定位

到您的站点时没有指定特定页的情况下为用户提供的页。在本项目中创建一个名为 Default.aspx 的页，并将它保存在站点的根文件夹中。如果用户在定位到站点时没有指定特定页，通过配置的应用程序，就可自动请求 Default.aspx 页。

2. 网站文件类型

ASP.NET 网站可以包含很多文件类型，这些文件中有很多可以由 ASP.NET 支持和管理。大多数 ASP.NET 文件类型都可以在 VS 开发环境的“添加新项”对话框中自动生成。ASP.NET 管理的文件类型能够在 ASP.NET 应用程序中被 ASP.NET 应用程序的不同模块访问和调用，这些文件可能是用户能够直接访问的，也有可能是用户无法直接访问的。ASP.NET 管理的常见文件类型及说明如表 5-10 所示。

表 5-10 ASP.NET 管理的常见文件类型及说明

文件类型	文件位置	功能说明
.asax	应用程序根目录	通常是 Global.asax 文件，该文件包含从 HttpApplication 类派生并表示该应用程序的代码
.ascx	应用程序根目录或子目录	Web 用户控件文件，该文件定义自定义、可重复使用的用户控件
.asmx	应用程序根目录或子目录	XML Web services 文件，该文件包含通过 SOAP 方式可用于其他 Web 应用程序的类和方法
.aspx	应用程序根目录或子目录	ASP.NET Web 窗体文件，该文件可包含 Web 控件和其他业务逻辑
.ashx	应用程序根目录或子目录	一般处理程序文件，该文件包含用于实现 IHttpHandler 接口的代码。
.browser	App_Browsers 子目录	浏览器定义文件，用于标识客户端浏览器的启用功能
.compile	Bin 子目录	预编译的存根文件，该文件指向相应的程序集。可执行文件类型（.aspx、ascx、.master、主题文件）已经过预编译并放在 Bin 子目录下
.config	应用程序根目录或子目录	通常是 Web.config 配置文件，该文件包含其设置配置各种 ASP.NET 功能的 XML 元素
.cs	App_Code 子目录；但如果是 ASP.NET 页的代码隐藏文件，则与网页位于同一目录	运行时要编译的类源代码文件。类可以是 HTTP 模块、HTTP 处理程序，或者是 ASP.NET 页 HTTP 处理程序介绍的代码隐藏文件
.csproj	Visual Studio 项目目录	Visual Studio 客户端应用程序项目的项目文件
.disco .vsdisco	App_WebReferences 子目录	XML Web Services 发现文件，用于帮助定位可用的 Web Services
.dll	Bin 子目录	已编译的类库文件，或者可以将类的源代码放在 App_Code 子目录下
.master	应用程序根目录或子目录	母版页，它定义应用程序中引用母版页的其他网页的布局
.mdb	App_Data 子目录	Access 数据库文件
.mdf	App_Data 子目录	SQL 数据库文件
.resources .resx	App_GlobalResources 或 App_LocalResources 子目录	资源文件，该文件包含指向图像、可本地化文本或其他数据的资源字符串
.sitemap	应用程序根目录	站点地图文件，该文件包含网站的结构。ASP.NET 中附带了一个默认的站点地图提供程序，它使用站点地图文件可以很方便地在网页上显示导航控件
.skin	App_Themes 子目录	用于确定显示格式的外观文件
.sln	Visual Web Developer 项目目录	Visual Web Developer 项目的解决方案文件
.soap	应用程序根目录或子目录	SOAP 扩展文件

除了上面给出的由 ASP.NET 管理的文件类型之外，还有一些是注册到由 IIS 管理的静态文件，如.css、.html。

5.2.3 ASP.NET 的应用程序文件夹及网站路径

一、实战演练

（1）在“解决方案资源管理器”中，右键单击项目名称，在弹出的快捷菜单中选择“添加 ASP.NET 文件夹”下的“App_Code”选项。

（2）右击“App_Code”，在弹出的快捷菜单中选择“添加新项”选项，在“添加新项”对话框中选择“类”选项。在“名称”框中输入类名“Common.cs”，该类主要用于整个项目中的数字校验。

（3）系统通用类 Common 如代码清单 5-2 所示。

代码清单 5-2 系统通用类 Common 代码

```
public static class Common
{
    //功能：判断字符串是否为整型
    public static bool IsInt(string str)
    {
        Regex regex = new Regex("^[0-9]*[1-9][0-9]*$");
        return regex.IsMatch(str.Trim());
    }
    //功能：判定查询字符串的有效性
    public static bool ValidQueryString(string str)
    {
        if (string.IsNullOrEmpty(str) || str.Length > 9 || !IsInt(str))
        {
            return false;
        }
        else
        {
            return true;
        }
    }
}
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-11 所示。

表 5-11 演练完成情况评价表

任 务 号	5-3	任 务 名 称	ASP.NET 网站中的共享代码文件夹
任务子项	完成情况	主要问题	未完成原因
添加 App_code 文件夹			
创建系统通用类 common			

三、知识点

1. ASP.NET 常见文件夹

为了将网站的文件保存在方便应用程序访问的文件夹中，ASP.NET 保留了某些可用于特

定类型的文件和文件夹名称，这些文件夹称为应用程序文件夹。ASP.NET 项目的常见文件夹如表 5-12 所示。

表 5-12 ASP.NET 常见文件夹

文件夹类型	说 明
App_Browsers	包含 ASP.NET 用于标识个别浏览器并确定其功能的浏览器定义（.browser）文件
App_Code	包含作为应用程序一部分进行编译的实用工具类和业务对象的源代码。在动态编译的应用程序中，当对应用程序发出首次请求时，ASP.NET 编译 App_Code 文件夹中的代码，然后在检测到任意更改时重新编译该文件夹中的项。在应用程序中将自动引用 App_Code 文件夹中的代码
App_Data	包含应用程序数据文件，包括 MDF 文件、XML 文件和其他数据存储文件。ASP.NET 2.0 使用 App_Data 文件夹来存储应用程序的本地数据库，该数据库可用于维护成员资格和角色信息
App_Themes	包含用于定义 ASP.NET 网页和控件外观的文件集合，如.skin 和 .css 文件及图像文件和一般资源
Bin	包含在应用程序中引用的控件、组件或其他代码的已编译程序集（.dll 文件）。在应用程序中将自动引用 Bin 文件夹中的代码所表示的任意类

在对网站文件及文件进行设置时，要注意不同用途的文件放在各自的文件夹中。用户自定义的一些文件或文件夹不要存放在这些指定的专业文件夹中，如图片文件，存放在上述任意文件夹中，在设计页面中都可以显示，但在进行程序调试浏览时就可能会出现不能正常显示的图片。

2. ASP.NET 网站路径

使用网站中的资源时，必须指定资源的路径。一般在网站中会使用 URL 路径引用页面中的图像文件或网站中其他位置处的页面的 URL。同样，Web 应用程序中的代码可以使用基于服务器的文件的物理文件路径对文件进行读写操作。ASP.NET 提供用于引用资源并确定应用程序中的页面或其他资源的路径的方法。

很多情况下，页面中的元素或控件必须引用外部资源，如文件。ASP.NET 支持引用外部资源的各种方法。根据您使用的是客户端元素还是 Web 服务器控件，选择的引用方法有所不同。

客户端元素是页面上的非 Web 服务器控件元素，它们将按原样被传递给浏览器。因此，从客户端元素中引用资源时，应根据 HTML 中 URL 的标准规则构造路径。以页面元素 img 的 src 属性为例：

绝对 URL 路径引用

```
<img src="http://www.163.com/ Images/SampleImage.jpg" />
```

网站根目录相对路径

```
<img src="/Images/SampleImage.jpg" />
```

根据当前页面路径解析的相对路径

```
<img src="Images/SampleImage.jpg" />  <img src="../Images/SampleImage.jpg" />
```

ASP.NET 服务器控件也使用绝对路径或相对路径，但是会存在不容易维护的问题，因此 ASP.NET 根目录运算符 (~)解析为当前应用程序的根目录。可以结合使用 ~ 运算符和文件夹来指定基于当前根目录的路径。下面是 Image 服务器控件指定根目录相对路径的 ~ 运算符实例：

```
<asp:image runat="server" id="Image1"  ImageUrl="~/Images/SampleImage.jpg" />
```

服务器控件中的任何与路径有关的属性都使用 ~ 运算符，但~ 运算符只能为服务器控件识别，并且位于服务器代码中，不能将 ~ 运算符用于客户端元素。

5.2.4 Web.config 配置文件

Web.config 文件是一个 XML 文本文件，用来储存 ASP.NET Web 应用程序的配置信息，它可以出现在应用程序的每一个目录中。当新建一个 Web 应用程序时，默认情况下会在根目录自动创建一个默认的 Web.config 文件。

一、实战演练

（1）在站点根目录下添加配置文件 Web.config。

（2）打开 Web.config 配置连接数据库的字符串，配置代码如下：

```
    <Configuration>
      <appSettings>
           <add key="Conn" value="DataSource=.\SQLEXPRESS; AttachDbFilename=
|DataDirectory |Enterprise.mdf; Integrated Security=True;Connect Timeout=30;
User Instance=True"/>
    </appSettings>
    ...
    </Configuration>
```

（3）在 Web.config 文件中，按如下方法配置 Session 状态：

```
    <Configuration>
      ...
    <system.web>
    ...
    <sessionState mode="InProc" cookieless="AutoDetect" timeout="30"/>

    </system.web>
      ...
    </Configuration>
```

（4）在 Web.config 文件中，添加节点，配置自定义错误。

```
    <Configuration>
      ...
    <system.web>
    ...
       <customErrors mode="On" defaultRedirect="GenericErrorPage.htm">
          <error statusCode="403" redirect="NoAccess.htm" />
          <error statusCode="404" redirect="FileNotFound.htm" />
       </customErrors>
    </system.web>
    ...
    </Configuration>
```

其中，NoAccess.htm 和 FileNotFound.htm 为事先在站点根目录下创建的两个 HTML 文件，分别用于提示无权限访问和页面文件不存在的错误信息。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-13 所示。

表 5-13 演练完成情况评价表

任务号	5-4	任务名称	网站常规配置
完成情况		主要问题	未完成原因

三、知识点

1. 配置概述

ASP.NET 的配置是一个易操作且功能强大的基于 XML 的配置系统。ASP.NET 的配置系统可以配置整个服务器上的所有 ASP.NET 应用程序、单个 ASP.NET 应用程序、各个页面或应用程序子目录；可以配置各种功能，如身份验证模式、页缓存、编译器选项、自定义错误、调试、跟踪选项等。ASP.NET 配置数据存储在全部命名为 Web.config 的 XML 文本文件中，Web.config 文件可以出现在 ASP.NET 应用程序的多个目录中。使用这些文件，可以在将应用程序部署到服务器上之前、期间或之后方便地编辑配置数据。

ASP.NET 配置系统支持如下两类配置文件：

1）服务器配置

服务器配置信息存储在一个名为 machine.config 的文件中。该文件描述了所有 ASP.NET Web 应用程序所用的默认设置。ASP.NET 会将一个 machine.config 文件安装到服务器上，可以在[WinNT]\Microsoft.NET\Framework\[version]\Config 下看到该文件。

2）应用程序配置

应用程序配置信息存储在一个名为 web.config 的文件中。该配置文件描述了一个单独的 ASP.NET 应用程序的设置信息。一个服务器可以有多个 web.config 文件，每个文件都可放在应用程序的根目录下或者放在应用程序内部的目录中。web.config 文件中所做的设置会覆盖 machine.config 提供的默认配置信息中的设置，或者是在其中添加新的设置。

2. web.config 配置文件

web.config 配置文件是基于 XML 文件类型的文件，所以 web.config 文件同样包含 XML 结构中的树形结构。在 ASP.NET 应用程序中，所有的配置信息都存储在 web.config 文件中的 configuration 配置节中。在此配置节中，包括配置节处理应用程序声明，以及配置节设置两个部分，其中，对处理应用程序的声明存储在 configSections 配置节内，示例代码如代码清单 5-3 所示。

代码清单 5-3 configSections 配置节示例代码

```
    <configSections>
       <sectionGroup name="system.web.extensions"
    type="System.Web.Configuration.SystemWebExtensionsSectionGroup,
                   System.Web.Extensions, Version=3.5.0.0,
                   Culture=neutral, PublicKeyToken=31BF3856AD364E35">
            <sectionGroup name="scripting"
                   type="System.Web.Configuration.ScriptingSectionGroup,
                   System.Web.Extensions, Version=3.5.0.0,
                  Culture=neutral, PublicKeyToken=31BF3856AD364E35">
                <section name="scriptResourceHandler"
                  type="System.Web.Configuration.ScriptingScriptResource
HandlerSection,
                   System.Web.Extensions, Version=3.5.0.0,
                 Culture=neutral, PublicKeyToken=31BF3856AD364E35"
                  requirePermission="false"
allowDefinition="MachineToApplication"/>
```

```
        <sectionGroup    name="webServices"
type="System.Web.Configuration.ScriptingWebServicesSectionGroup,
              System.Web.Extensions, Version=3.5.0.0,
               Culture=neutral, PublicKeyToken=31BF3856AD364E35">
        </sectionGroup>
    </sectionGroup>
  </sectionGroup>
</configSections>
```

配置节设置区域中的每个配置节都有一个应用程序声明。节处理程序是用来实现ConfigurationSection 接口的.NET Framework 类。节处理程序声明中包括了配置设置节的名称，以及用来处理该配置节中的应用程序的类名。

配置节设置区域位于配置节处理程序声明区域之后。对配置节的设置还包括子配置节的配置，这些子配置节同父配置节一起描述一个应用程序的配置，通常情况下这些配置节由同一个配置节进行管理，示例代码如代码清单 5-4 所示。

代码清单 5-4　web.config 节点配置示例代码

```
<pages>
   <controls>
      <add tagPrefix="asp" namespace="System.Web.UI"
         assembly="System.Web.Extensions,
         Version=3.5.0.0, Culture=neutral, PublicKeyToken=31BF3856AD 364E35"/>
        <add tagPrefix="asp" namespace="System.Web.UI.WebControls" assembly="
        System.Web.Extensions,
   </controls>
</pages>
```

虽然 web.config 配置文件是基于 XML 文件格式的，但是在 web.config 配置文件中并不能随意地自行添加配置节或者修改配置节的位置，如 pages 配置节就不能存放在 configSections 配置节之中。在创建 Web 应用程序时，系统通常会在文件中自行创建一个 web.config 配置文件，并且通常已经规定好了该配置文件的结构。

3．web.config 配置文件中基本的配置节

在 web.config 配置文件中包括很多配置节，这些配置节都用来规定 ASP.NET 应用程序的相应属性。

1）configuration 根配置节

所有 web.config 的根配置节都存储在 configuration 标记中，在它内部封装了其他的配置节，示例代码如下所示：

```
<configuration>
        <system.web>
        ……
        </system.web>
</configuration>
```

2）configSections 处理声明配置节

该配置节主要用于自定义的配置节处理程序声明，由多个 section 配置节组成，示例代码如代码清单 5-3 所示。其中，section 配置节包括 name 和 type 两种属性，name 属性指定配置数据配置节的名称，而 type 属性指定与 name 属性相关的配置处理程序类。

3）appSettings 用户自定义配置节

appSettings 配置节为开发人员提供 ASP.NET 应用程序的扩展配置，通过使用 appSettings

配置节能够自定义配置文件，示例代码如下所示：

```
<appSettings>
    <add key="Name" value="Guojing"/>    //增加自定义配置节
    <add key="E-mail" value="soundbbg@live.cn"/>
</appSettings>
```

上述代码添加了两个自定义配置节，这两个自定义配置节分别为 Name 和 E-mail，用于定义该 Web 应用程序开发者的信息，以便在其他页面中使用。

若需要在页面中使用该配置节，可以使用 ConfigurationSettings.AppSettings(Key 的名称)方法来获取自定义配置节中的配置值，示例代码如下所示：

```
protected void Page_Load(object sender, EventArgs e)
{  //获取自定义配置节
   TextBox1.Text = ConfigurationSettings.AppSettings["name"].ToString();
}
```

appSettings 配置节包括两个属性，分别为 Key 和 Value。Key 属性指定自定义属性的关键字，以方便在应用程序中使用该配置节，而 Value 属性则定义该自定义属性的值。

4）customErrors 用户错误配置节

该配置节能够指定当出现错误时，系统自动跳转到一个错误发生的页面，同时也能够为应用程序配置是否支持自定义错误。customErrors 配置节包括两种属性，分别为 mode 和 defaultRedirect。其中 mode 包括 3 种状态，分别为 On、Off 和 RemoteOnly，On 表示启动自定义错误；Off 表示不启动自定义错误；RemoteOnly 表示给远程用户显示自定义错误。另外 defaultRedirect 属性则配置了当应用程序发生错误时跳转的页面。

customErrors 配置节还包括子配置节 error，该标记用于特定状态的自定义错误页面。子标记 error 包括两个属性，分别为 statusCode 和 redirect，其中 statusCode 用于捕捉发生错误的状态码，而 redirect 指定发生该错误后跳转的页面。该配置节的配置代码如下所示：

```
<customErrors mode="RemoteOnly" defaultRedirect="GenericErrorPage.htm">
    <error statusCode="403" redirect="NoAccess.htm" />
    <error statusCode="404" redirect="FileNotFound.htm" />
</customErrors>
```

上述代码在 web.config 文件中配置了相应的 customErrors 信息。当出现 404 错误时，系统会自行跳转到 FileNotFound.htm 页面以提示 404 错误，开发人员能够编写 FileNotFound.htm 页面进行用户提示。

5）globalization 全局编码配置节

globalization 用于配置应用程序的编码类型，ASP.NET 应用程序将使用该编码类型分析 ASPX 等页面。

在配置 globalization 配置节时，其编码类型可以参考上述编码类型，如果不指定编码类型，则 ASP.NET 应用程序默认编码为 UTF-8，示例代码如下所示：

```
<globalization fileEncoding="UTF-8" requestEncoding="UTF-8" responseEncoding="UTF-8"/>
```

6）sessionState 状态配置节

sessionState 配置节用于完成 ASP.NET 应用程序中会话状态的设置，包括以下 5 种属性：

✧ mode：指定会话状态的存储位置，有 Off、Inproc、StateServer 和 SqlServer 集中设置，

其中，Off 表示禁用该设置；Inproc 表示在本地保存会话状态；StateServer 表示在服务器上保存会话状态；SqlServer 表示在 SQL Server 保存会话设置。

✧ stateConnectionString：用来指定远程存储会话状态的服务器名和端口号。

✧ sqlConnectionString：用来连接 SQL Server 的连接字符串，当在 mode 属性中设置 SqlServer 时，则需要使用该属性。

✧ Cookieless：指定是否使用客户端 Cookie 保存会话状态。

✧ Timeout：指定在用户无操作时超时的时间，默认情况为 20 分钟。

sessionState 配置节的示例代码如下所示：

```
<sessionState mode="InProc" timeout="25" cookieless="false"></sessionState>
```

ASP.NET 不仅包括这些基本的配置节，还包括其他高级的配置节。高级的配置节通常用于指定界面布局样式，如母版页、默认皮肤、伪静态等高级功能。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 1

拓展任务号	5-1	任务名称	配置 Web.config
计划用时	15 分钟	任务性质	课内
任务描述与目标			
在页面中为了实现代码重用，常会用到用户控件，但如果网站的许多页面上使用控件或移动了.ascx 文件，就需要更新所有的注册声明，这样带来了很多不便。通过配置 Web.config 文件，可以避免在页面上重复声明控件			
主要操作步骤提示			

1．打开项目，新一个 Controls 文件夹，新建两个用户控件 Header.ascx 和 Footer.ascx；
2．打开 Web.config 文件，进行修改，参考代码如下：
`<pages>`
`  <controls>`
`  <add tagPrefix="scottgu"`
`  src="~/Controls/Header.ascx" tagName="header"/>`
`  <add tagPrefix="scottgu"`
`  src="~/Controls/Footer.ascx" tagName="footer"/>`
`  <add tagPrefix="ControlVendor" assembly="ControlVendorAssembly"/>`
`  </controls>`
`</pages>`
3．配置后将控件放置在项目设置目录中即可，新建 Web 页，测试操作的正确性			

5.3 任务 2 ASP.NET 网站生命周期与状态管理

任务描述

ASP.NET 网站从网站启动到网站停止运行是一个网站的生命周期，在生命周期不同阶段会产生一系列的事件。为了管理生命周期，ASP.NET 提供了系列的类、事件。

Web 应用程序的本质上是无状态的。每次将网页发送到服务器时，都会创建网页类的一

个新实例，而且默认情况下，来自一个请求的信息对下一个请求是不可用的。这通常意味着在每一次往返行程中，与该页及该页上的控件相关联的所有信息都会丢失。如果要想生成需要维护某些跨请求状态信息的 Web 应用程序（如实现购物车、数据滚动等的应用程序），就可能非常困难。为了克服这种限制，ASP.NET 采用了管理状态的功能用于存储请求之间的信息。应用程序状态、会话状态和 Cookie 是最常用的方法。应用程序状态、会话状态的开始和结束会触发一些相关事件，这些事件由 Global.asax 文件来负责管理。

通过本任务主要学习 ASP.NET 网站生命周期及状态管理所涉及的类、事件。

解决方案

为完成本任务，要完成以下几个方面的工作：

1．能使用 Global.asax 文件；

2．能创建 ASP.NET 应用程序级别的事件处理程序；

3．能使用综合应用程序变量 Application 对象和会话变量 Session 对象；

4．能使用 Cookie 对象。

5.3.1 应用程序生命周期与 Global.asax 文件

在应用程序的生命周期期间，应用程序会引发可处理的事件并调用可重写的特定方法。处理应用程序事件或方法由应用程序根目录中的 Global.asax 文件来统一处理，一个应用程序文件只有一个 Global.asax，只能放在应用程序根目录下，不能重命名该文件。

一、实战演练

（1）打开 EnterPriseWeb 网站，右击项目名，选择“添加”→“添加新项”，在如图 5-3 所示的添加新项对话框中选择“全局应用程序类”。

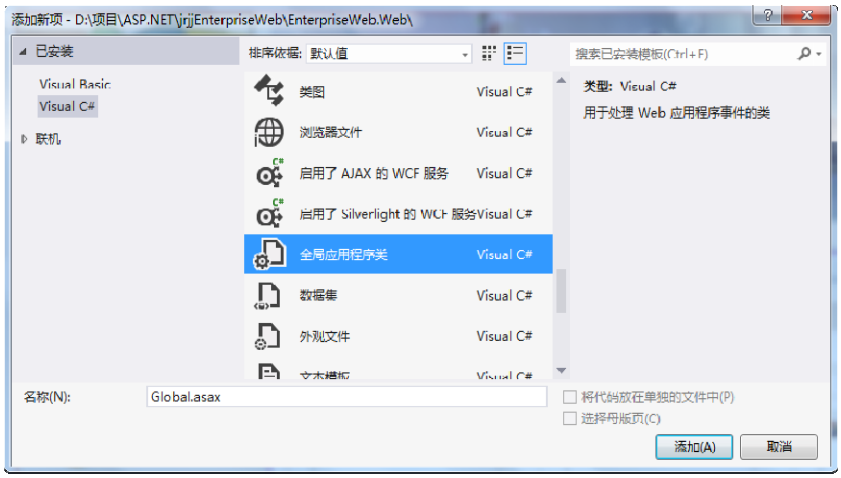

图 5-3 添加 “全局应用程序类”对话框

（2）单击“添加”按钮，在“解决方案”面板中可以看到新添加的 Global.asax 文件，打开 Global.asax 文件，可以看到其中自动生成的事件代码。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-14 所示。

表 5-14　演练完成情况评价表

任务号	5-5	任务名称	添加全局应用程序类
任务子项	完成情况	主要问题	未完成原因
在应用程序项目中添加全局应用程序类并观察其代码格式。			

三、知识点

1. 应用程序生命周期

ASP.NET 的应用程序生命周期与 Web 服务器相关，不同的 IIS 服务器下 ASP.NET 的应用程序生命周期略有不同。IIS 7.0 和 IIS 6.0 的处理阶段之间的主要区别在于 ASP.NET 如何与 IIS 服务器集成。在 IIS 6.0 中，有两个请求处理管道。一个管道用于本机代码 ISAPI 筛选器和扩展组件。另一个管道用于托管代码应用程序组件。在 IIS 7.0 中，ASP.NET 运行库与 Web 服务器集成，这样就有了一个针对所有请求的统一请求处理管道。下面列出了 IIS 7.0 集成模式下运行的 ASP.NET 应用程序生命周期的各个阶段。

1）发出一个对应用程序资源的请求

ASP.NET 应用程序的生命周期以浏览器向 Web 服务器发送请求为起点。

2）统一管道接收对应用程序的第一个请求

当统一管道接收对应用程序中的任何资源的第一个请求时，将为 ApplicationManager 类创建一个实例，该实例就是处理请求的应用程序域。在第一个请求期间，如果需要，将对应用程序中的顶级项进行编译，其中包括 App_Code 文件夹中的应用程序代码。在应用程序域中，将为 HostingEnvironment 类创建一个实例，该实例提供对有关应用程序的信息（如存储该应用程序的文件夹的名称）的访问。

3）为每个请求创建响应对象

在创建了应用程序域并对 HostingEnvironment 对象进行了实例化之后，将创建并初始化应用程序对象，如 HttpContext、HttpRequest 和 HttpResponse。HttpContext 类包含特定于当前应用程序请求的对象，如 HttpRequest 和 HttpResponse 对象。HttpRequest 对象包含有关当前请求的信息，包括 Cookie 和浏览器信息。HttpResponse 对象包含发送到客户端的响应，其中包括所有呈现的输出和 Cookie。

4）将 HttpApplication 对象分配给请求

初始化所有应用程序对象之后，将通过创建 HttpApplication 类的实例来启动应用程序。如果应用程序有 Global.asax 文件，则 ASP.NET 会创建从 HttpApplication 类派生的 Global.aspx 类的实例。然后使用该派生类来表示应用程序。

5）由 HttpApplication 管线处理请求

在处理请求时，HttpApplication 类会执行系统任务。这些事件对于希望在引发关键请求管道事件时运行代码的网页开发人员很有用。

2. ASP.NET 页生命周期

ASP.NET 页运行时，此页将经历一个生命周期，在生命周期中将执行一系列处理步骤。

这些步骤包括初始化、实例化控件、还原和维护状态、运行事件处理程序代码以及进行呈现。了解页生命周期非常重要，因为这样做您就能在生命周期的合适阶段编写代码，以达到预期效果。此外，如果您要开发自定义控件，就必须熟悉页生命周期，以便正确进行控件初始化，使用视图状态数据填充控件属性以及运行任何控件行为代码。在页生命周期的每个阶段中，页将引发可运行您自己的代码进行处理的事件。

1）页请求

页请求发生在页生命周期开始之前。用户请求页时，ASP.NET 将确定是否需要分析和编译页（从而开始页的生命周期），或者是否可以在不运行页的情况下发送页的缓存版本以进行响应。

2）开始

在开始阶段，将设置页属性，如 Request 和 Response。在此阶段，页还将确定请求是回发请求还是新请求，并设置 IsPostBack 属性。

3）页初始化

页初始化期间，可以使用页中的控件，并将设置每个控件的 UniqueID 属性。此外，任何主题都将应用于页。如果当前请求是回发请求，则回发数据尚未加载，并且控件属性值尚未还原为视图状态中的值。

4）加载

加载期间，如果当前请求是回发请求，则将使用从视图状态和控件状态恢复的信息加载控件属性。

5）验证

在验证期间，将调用所有验证程序控件的 Validate 方法，此方法将设置各个验证程序控件和页的 IsValid 属性。

6）回发事件处理

如果请求是回发请求，则将调用所有事件处理程序。

7）呈现

在呈现之前，会针对该页和所有控件保存视图状态。在呈现阶段中，页会针对每个控件调用 Render 方法，它会提供一个文本编写器，用于将控件的输出写入页的 Response 属性的 OutputStream 中。

8）卸载

完全呈现页并已将页发送至客户端、准备丢弃该页后，将调用卸载。此时，将卸载页属性（如 Response 和 Request）并执行清理。

3. Global.asax 文件

Global.asax 文件也称为 ASP.NET 应用程序文件，它一般被放在根目录下。此文件中的代码不产生用户界面，也不响应单个页面的请求。Global.asax 文件是可选的，只在处理应用程序事件或会话事件时才创建。

Global.asax 文件包含的常用事件如下：

✧ Application_Init：在应用程序被实例化或第一次被调用时，该事件被触发，对于所有的 HttpApplication 对象实例，它都会被调用；

✧ Application_Error：当应用程序中遇到一个未处理的异常时，该事件被触发；

✧ Application_Start：在 HttpApplication 类的第一个实例被创建时，该事件被触发，它允许创建可以由所有 HttpApplication 实例访问的对象；

✧ Application_End：在 HttpApplication 类的最后一个实例被销毁时，该事件被触发，在一个应用程序的生命周期内它只被触发一次；

✧ Session_Start：在一个新用户访问应用程序 Web 站点时，该事件被触发；

✧ Session_End：在一个用户的会话超时、结束或离开应用程序 Web 站点时，该事件被触发。

5.3.2 ASP.NET Application 应用程序对象

应用程序状态是可供 ASP.NET 应用程序中的所有类使用的数据储存库。它存储在服务器的内存中，因此与在数据库中存储和检索信息相比，它的执行速度更快。与特定于单个用户会话的会话状态不同，应用程序状态应用于所有的用户和会话。因此，应用程序状态非常适合存储那些数量少、不随用户的变化而变化的常用数据。

一、实战演练

（1）在企业网站项目中添加新项，选择“Web 用户控件”，命名为“Counter.ascx”。

（2）切换到“源”视图中，添加如下代码：

```
<font face="宋体">您是第<%=Application["counter"]%> 位访问者！目前在线人数<%=Session["counter"]%> </font>
```

（3）在企业网站项目中打开 Global.asax 文件，添加相关代码，如代码清单 5-5 所示。

代码清单 5-5 Global.asax 文件代码段

```
<script runat="server">
    void Application_Start(object sender, EventArgs e)
    {
        //在应用程序启动时运行的代码
        uint count = 0;
        StreamReader srd;
        //取得文件的实际路径
        string file_path = Server.MapPath("counter.txt");
        //打开文件进行读取
        srd = File.OpenText(file_path);
        while (srd.Peek() != -1)
        {
            string str = srd.ReadLine();
            count = UInt32.Parse(str);
        }
        object obj = count;
        Application["counter"] = obj;
        srd.Close();
    }
    void Application_End(object sender, EventArgs e)
    {
        //在应用程序关闭时运行的代码
        //装箱
        uint js = 0;
```

```
        js = (uint)Application["counter"];
        //将数据记录写入文件
        string file_path = Server.MapPath("counter.txt");
        StreamWriter fs = new StreamWriter(file_path, false);
        fs.WriteLine(js);
        fs.Close();
    }
      void Session_Start(object sender, EventArgs e)
    {
        //会话开始，添加在线人数
        Session["counter"]= Convert.ToInt32 (Session["counter"])+1;
        //在新会话启动时运行的代码
        Application.Lock();
        //数值累加
        Application["counter"] =Convert.ToInt32 (Application["counter"])+1;
        //将数据记录写入文件
        string file_path = Server.MapPath("counter.txt");
        StreamWriter fs = new StreamWriter(file_path, false);
        fs.WriteLine(jishu);
        fs.Close();
        Application.UnLock();
    }
    void Session_End(object sender, EventArgs e)
    {
        //会话结束，即用户退出，减去在线人数
        Session["counter"]= Convert.ToInt32 (Session["counter"])-1;
    }
</script>
```

（4）在母版页中，在特定的位置上插入刚创建的计数器用户控件。

（5）运行测试。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-15 所示。

表 5-15 演练完成情况评价表

任 务 号	5-6	任 务 名 称	网站前台页面设计
任务子项	完成情况	主要问题	未完成原因
在页面中添加简单的计数器			

三、知识点

应用程序状态存储在 HttpApplicationState 类的实例 Application 对象中。该类公开对象的键值字典。在用户初次访问应用程序中的任何 URL 资源时，将会创建 HttpApplicationState 实例。HttpApplicationState 类通常通过 HttpContext 类的 Application 属性访问。

1. Application 对象

Application 对象可以在多个请求、连接之间共享公用信息，也可以充当信息传递的管道。Application 对象来自 HttpApplicationStat 类，该类启用 ASP.NET 应用程序中多个会话和请求

之间的全局信息共享。

HttpApplicationState 类常见的属性和方法如下：

✧ Count 属性：获取 HttpApplicationState 集合中的对象数；

✧ Item [([(Int32)])]属性：通过索引获取单个 HttpApplicationState 对象；

✧ Item [([(String)])]属性：通过名称获取单个 HttpApplicationState 对象的值；

✧ Add()方法：将新的对象添加到 HttpApplicationState 集合中；

✧ Clear()方法：从 HttpApplicationState 集合中移除所有对象；

✧ Lock()方法：锁定对 HttpApplicationState 变量的访问以促进访问同步；

✧ Remove()方法：从 HttpApplicationState 集合中移除命名对象；

✧ RemoveAll()方法：从 HttpApplicationState 集合中移除所有对象；

✧ RemoveAt()方法：按索引从集合中移除一个 HttpApplicationState 对象；

✧ UnLock()方法：取消对 HttpApplicationState 变量的锁定访问以促进访问同步。

如下示例代码用锁定方法将值写入应用程序状态：

```
Application.Lock();
Application["PageRequestCount"] = ((int)Application["PageRequestCount"])+1;
Application.UnLock();
```

2. Application 和 Session 对象的比较

Session 对象与 Application 对象有些相似，只是作用域不同。Application 对象是针对所有用户，而 Session 对象则相反，每个用户都有自己的 Session 对象，它的生命周期起始于服务器产生对用户请求页面的响应，终止于用户断开与服务器的连接。Application 对象不会像 Session 对象那样当一个新用户请求就触发事件，Application 对象的事件只触发一次，就是在第一个用户的第一个请求时。一个 Application_End 事件肯定发生在 Session_End 事件之后，只有在服务器停止工作或 Application_End 事件卸载时才触发。

5.3.3 ASP.NET Cookie 应用

Cookie 是一小段文本信息，伴随着用户请求和页面在 Web 服务器和浏览器之间传递。Cookie 包含每次用户访问站点时 Web 应用程序都可以读取的信息。

一、实战演练

（1）在企业网站项目打开 Web.master 母版页，在页面的左下方添加如下代码

```
<p>
    <asp:Label ID="lbDate" runat="server" Text="Label"></asp:Label>
 </p>
```

（2）切换 Web.master 的代码视图，在 Page_Load 事件添加如下代码：

```
if (Request.Cookies["UserSettings"] != null)
  {
     Label1.Text = "您上次光临的时间是："
+ Server.HtmlEncode(Request.Cookies["UserSettings"]["date"]);
  }
  else
   {
```

```
        HttpCookie myCookie = new HttpCookie("UserSettings");
        myCookie["date"] = DateTime.Now.ToString("d");
       Response.Cookies["UserSettings"].Expires = DateTime.Now.AddDays(30d);
        Response.Cookies.Add(myCookie);
        Label1.Text = "欢迎初次光临！";
    }
```

（3）对页面多运行测试变化。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-16 所示。

表 5-16　演练完成情况评价表

任　务　号	5-7	任 务 名 称	给网站添加 Cookie
任务子项	完成情况	主要问题	未完成原因
母版页中添加用户访问记录			

三、知识点

Cookie 对象是一小段文本信息，伴随着用户请求和页面在 Web 服务器和浏览器之间进行传递。Cookie 对象包含每次用户访问站点时 Web 应用程序都可以读取的信息。

HttpCookie 类常见的属性如下：

✧ Expires 属性：获取或设置此 Cookie 的过期日期和时间；

✧ HasKeys 属性：获取一个值，通过该值指示 Cookie 是否具有子键；

✧ Name 属性：获取或设置 Cookie 的名称；

✧ Path 属性：获取或设置要与当前 Cookie 一起传输的虚拟路径；

✧ Value 属性：获取或设置单个 Cookie 值；

✧ Values 属性：获取单个 Cookie 对象所包含的键值对的集合。

如下几个小实例展示了 Cookie 对象的使用。

示例 1：代码示例演示了一个名为 myCookie 的 HttpCookie 对象的实例，该实例表示一个名为 UserSettings 的 Cookie，该 Cookie 的有效期是 1 天。

```
Response.Cookies["UserSettings"]["Font"] = "Arial";
Response.Cookies["UserSettings"]["Color"] = "Blue";
Response.Cookies["UserSettings"].Expires = DateTime.Now.AddDays(1d);
```

示例 2：读取名为 UserSettings 的 Cookie，然后读取名为 Font 子键的值。

```
if (Request.Cookies["UserSettings"] != null)
  {
    string userSettings;
      if (Request.Cookies["UserSettings"]["Font"] != null)
        { userSettings = Request.Cookies["UserSettings"]["Font"]; }
      }
```

示例 3：通过为 Cookie 设置已过去日期的方式删除 Cookie。

```
if (Request.Cookies["UserSettings"] != null)
```

```
{
 HttpCookie myCookie = new HttpCookie("UserSettings");
 myCookie.Expires = DateTime.Now.AddDays(-1d);
 Response.Cookies.Add(myCookie);
}
```

5.4 任务 3 ASP.NET 的母版页与导航技术

任务描述

ASP.NET 母版页使您可以创建页面布局（母版页），而您可以对网站中的选定页或所有页（内容页）使用该页面布局。母版页可以极大地简化为站点创建一致外观的任务。本节中的主题介绍母版页，并描述如何创建和管理它们。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）前后台母版页的设计和基于母版页的内容页设计；

（2）主题与外观的使用，更高效地维护网站的样式；

（3）网站导航设计。

5.4.1 ASP.NET 的母版页与皮肤

一、实战演练

（1）在“解决方案资源管理器”中，右键单击该项目的名称，在弹出的快捷菜单中选择“添加新项”选项。然后，在“模板列表”中选择“母版页”类型，将其命名为“Web.master”。

（2）进行母版页的页面布局，母版页的设计效果如图 5-4 所示。

图 5-4 母版页的设计效果

（3）母版页的关键源代码如代码清单 5-6 所示。

代码清单 5-6 母版页的关键代码

```
<!--Content-->
<div class="content">
  <div class="introImage">
  <asp:Image ID="imgBanner" ImageUrl="~/images/banner.gif" runat="server" />
  </div>
<!—可编辑区—>
    <asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
    </asp:ContentPlaceHolder>
</div>
<!--EndContent-->
<div class="footer">
<div class="copyright">
<a href="About.aspx">关于我们 About us </a></div>
</div>
</div>
```

（4）定义完母版页后，将向网站添加 ASP.NET 页面，这里要添加的是新闻详细页 ArticleShow. aspx。在“解决方案资源管理器”中，右键单击该项目的名称，在弹出的快捷菜单中选择“添加新项”选项。然后，在“模板列表”中选择“Web 窗体”类型，并为 ArticleShow.aspx 文件命名，同时，选中“选择母版页”复选框，单击“确定”按钮。

（5）在“选择母版页”对话框中，选择母版页所在的文件，然后选择此 ASP.NET 新页面应使用的母版页。“选择母版页”对话框如图 5-5 所示。

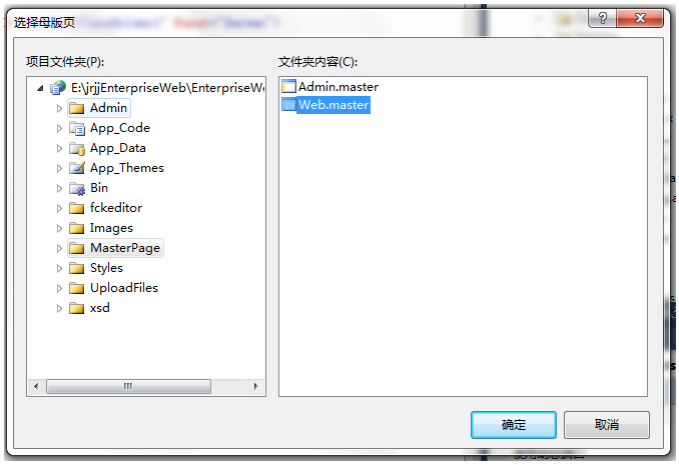

图 5-5 “选择母版页”对话框

（6）根据母版页新建的 ASP.NET 页面与直接创建的页面在结构上有很大的区别。页面中将包含以下标记，如代码清单 5-7 所示。

代码清单 5-7 套用母版页的子页关键代码

```
    <%@ Page Title="" Language="C#" MasterPageFile="~/MasterPage/Web.master"
AutoEventWireup= "true" CodeFile="Default5.aspx.cs" Inherits="Default5" %>
      <asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">
    </asp:Content>
    <asp:Content ID="Content2"
      ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">
    </asp:Content>
```

（7）将@Page 指令的 Title 属性设置为新闻页，并向 ID 为 Content2 的控件内添加 HTML 代码。

（8）新闻页的设计效果如图 5-6 所示。

图 5-6　新闻页的设计效果

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-17 所示。

表 5-17　演练完成情况评价表

任务号	5-8	任务名称	母版页设计与应用
任务子项	完成情况	主要问题	未完成原因
创建前台母版页			
应用前台母版页制作新闻页			

三、知识点

1. 母版页

1）母版页概述

母版页是扩展名为.master 的 ASP.NET 文件，它具有可以包括静态文本、HTML 元素和服务器控件的预定义布局。母版页由特殊的@ Master 指令识别，该指令替换了用于普通.aspx 页的@ Page 指令。

除@ Master 指令外，母版页还包含页的所有顶级 HTML 元素，如 html、head 和 form。如在母版页上可以将一个 HTML 表用于布局、将一个 img 元素用于公司徽标、将静态文本用于版权声明并使用服务器控件创建站点的标准导航。可以在母版页中使用任意 HTML 元素和 ASP.NET 元素。

除了在所有页上显示静态文本和控件外，母版页还包括一个或多个 ContentPlaceHolder 控件。ContentPlaceHolder 控件是可替换内容占位符，这些占位符控件定义可替换内容出现的区域，在内容页中定义可替换内容。ContentPlaceHolder 控件的语法如下：

```
<asp:ContentPlaceHolder id="ContentPlaceHolder1" runat="server">
</asp:ContentPlaceHolder>
```

2）内容页

通过创建各个内容页来定义母版页占位符控件的内容，这些内容页为绑定到特定母版页的 ASP.NET 页。通过包含指向要使用母版页的 MasterPageFile 属性，在内容页的 @Page 指令中建立绑定。

如一个内容页可能包含下面的 @ Page 指令，该指令将该内容页绑定到 Master1.master <%@ Page Language="C#" MasterPageFile="~/MasterPages/Master1.master" Title="Content Page"%>。

在内容页中，将通过添加自动 Content 控件并将这些控件映射到母版页上的 ContentPlaceHolder 控件来创建内容，如母版页可能包含名为 Main 和 Footer 的内容占位符。所以就会创建两个 Content 控件，一个映射到 ContentPlaceHolder 控件 Main，而另一个映射到 ContentPlaceHolder 控件 Footer。

3）替换占位符内容

创建 Content 控件后，向这些控件添加文本和控件。在内容页中，Content 控件外的任何内容（除服务器代码的脚本块外）都将导致错误。在 ASP.NET 页中执行的所有任务都可以在内容页中执行，如可以使用服务器控件和数据库查询或其他动态机制来生成 Content 控件的内容。替换占位符内容的具体过程如图 5-7 所示。

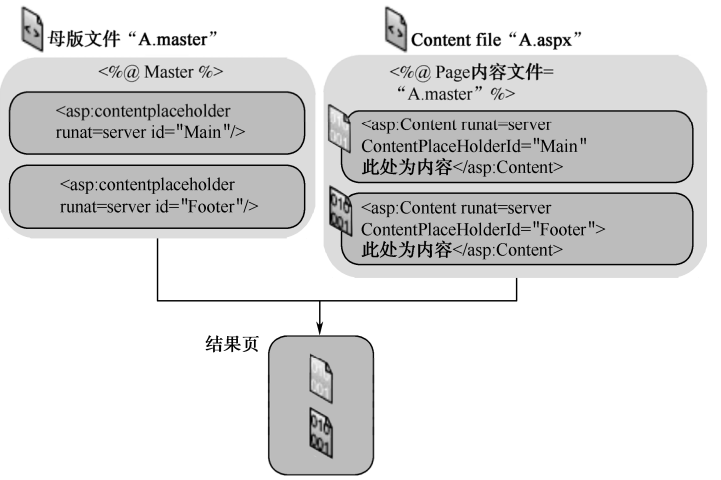

图 5-7　替换占位符内容的具体过程

2. 主题与外观

1）主题

主题是属性设置的集合，使用这些设置可以定义页面和控件的外观，然后在某个 Web 应用程序中的所有页、整个 Web 应用程序或服务器上的所有 Web 应用程序中一致地应用此外观。主题由一组元素组成，即外观、级联样式表（CSS）、图像和其他资源，至少包含外观元素。它是在网站或 Web 服务器上的特殊目录中定义的。

主题还可以包含图形和其他资源，如脚本文件或声音文件。例如，页面主题的一部分可能包括 TreeView 控件的外观，可以在主题中包括用于表示展开按钮和折叠按钮的图形。

2）外观

外观文件具有文件扩展名.skin，它包含各个控件（如 Button、Label、TextBox 或 Calendar

控件）的属性设置。控件外观设置类似于控件标记本身，但只包含要作为主题的一部分来设置的属性。如下是 Button 控件的控件外观：

```
<asp:button runat="server" BackColor="lightblue" ForeColor="black" />
```

在主题文件夹中创建.skin 文件。一个.skin 文件可以包含一个或多个控件类型的一个或多个控件外观，可以为每个控件在单独的文件中定义外观，也可以在一个文件中定义所有主题的外观。

有两种类型的控件外观，即默认外观和已命名外观。当向页应用主题时，默认外观自动应用于同一类型的所有控件。如果控件外观没有 SkinID 属性，则是默认外观；如果为 Calendar 控件创建一个默认外观，则该控件外观适用于使用本主题页面上的所有 Calendar 控件。已命名外观是设置了 SkinID 属性的控件外观。已命名外观不会自动按类型应用于控件，而应当通过设置控件的 SkinID 属性将已命名外观显式应用于控件。通过创建已命名外观，可以为应用程序中同一控件的不同实例设置不同的外观。

3）级联样式表

主题还可以包含级联样式表.css 文件。将 .css 文件放在主题文件夹中时，样式表会自动作为主题的一部分加以应用。使用文件扩展名 .css 在主题文件夹中定义样式表。

4）主题的应用范围

可以定义单个 Web 应用程序的主题，也可以定义供 Web 服务器上的所有应用程序使用的全局主题。定义主题之后，可以使用 @ Page 指令的 Theme 或 StyleSheetTheme 属性将该主题放置在个别页上；或者通过设置应用程序配置文件中的 pages 元素（ASP.NET 设置架构），将其应用于应用程序中的所有页。如果在 Machine.config 文件中定义了 pages 元素（ASP.NET 设置架构），则主题将应用于服务器上的 Web 应用程序中的所有页。

四、任务拓展

本节完成一个课内拓展实践任务。

拓展任务卡 2

拓展任务号	5-2	任务名称	创建与应用主题
计划用时	15 分钟	任务性质	课内
任务描述与目标			
通过本节练习掌握 ASP.NET 网站中的主题、外观的概念，掌握主题的创建方法和在网站中应用主题的方式			
主要操作步骤提示			

1．创建主题，主题名称为 mytheme，使用鼠标右键单击该主题，在弹出的快捷菜单中选择“添加新项”选项，依次添加外观 SkinFile.skin 和样式文件 StyleSheet.css。SkinFile.skin 文件代码如下所示：

```
<asp:HyperLink SkinId="hlDiff" runat="server"
  Font-Size="Small"  Font-Underline="false">
  </asp:Hyperlink>
<asp:Label SkinId="lblDiff" runat="server"  Text=""
  Font-Size="Small"></asp:Label>
```

2．应用主题，在本项目中所有的网页均应用同一主题，这需要设置 Web.config 文件中的 pages 配置节，如：<pages styleSheetTheme="mytheme"></pages>

5.4.2 ASP.NET 的站点导航技术

ASP.NET 站点导航功能为用户导航站点提供一致的方法。随着站点内容的增加及站点内来回移动网页，管理所有的链接可能会变得比较困难。ASP.NET 站点导航能够将指向所有页面的链接存储在一个中央位置，并在列表中呈现这些链接，或用一个特定的 Web 服务器控件在每页上呈现导航菜单。

一、实战演练

1. 创建站点地图

（1）在“解决方案资源管理器”中右键单击该项目的名称，在弹出的快捷菜单中选择“添加新项”选项。在“模板列表”中选择“站点地图”类型，将其命名为“Web.siteMap”。

（2）打开 Web.sitemap 文件，其初始内容如下：

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
       <siteMapNode url="" title=""  description="">
       <siteMapNode url="" title=""  description="" />
       <siteMapNode url="" title=""  description="" />
   </siteMapNode>
</siteMap>
```

（3）根据网站前台栏目结构，修改 Web.siteMap 文件内容，完成后的 Web.siteMap 代码如代码清单 5-8 所示。

代码清单 5-8 Web.siteMap 的代码

```
<?xml version="1.0" encoding="utf-8" ?>
  <siteMap xmlns="http://schemas.microsoft.com/AspNet/SiteMap-File-1.0" >
    <siteMapNode url="~/Default.aspx" title="首页"  description="">
       <siteMapNode url="~/About.aspx" title="关于我们"  description="" />
        <siteMapNode url="~/ArticleList.aspx" title="文章中心"  description="" >
            <siteMapNode url="~/ArticleShow.aspx" description=""title="正文
"/>
         </siteMapNode>
       <siteMapNode url="~/ProductList.aspx" title="产品中心"  description="" >
          <siteMapNode url="~/ProductShow.aspx" description="" title="产品
介绍"/>
        </siteMapNode>
       <siteMapNode url="~/Feedback.aspx" title="意见反馈"  description=""/>
   </siteMapNode>
</siteMap>
```

2. 添加 SiteMapPath 控件

打开 Web.master 母版页，在企业宣传图的下方添加 SiteMapPath 控件。添加控件后的 Web.master 源文件中增加了如下代码：

```
<asp:SiteMapPath ID="SiteMapPath1" runat="server">
        </asp:SiteMapPath>
```

3. 保存母版页 Web.master

在 VS 2012 中打开文章正文页 ArticleShow.aspx，页面显示如图 5-8 所示，比较与图 5-6 的不同之处。

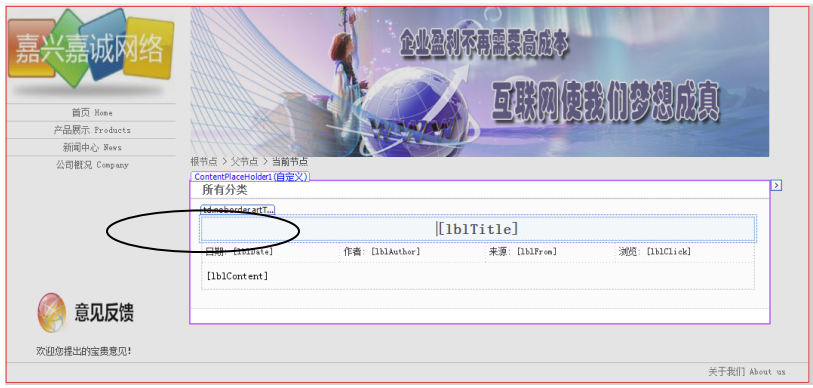

图 5-8　添加了 SiteMapPath 控件后的文章正文页

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-18 所示。

表 5-18　演练完成情况评价表

任务号	5-9	任务名称	使用 ASP.NET 的站点导航技术
任务子项	完成情况	主要问题	未完成原因
创建站点地图			
添加 SiteMapPath 控件			

三、知识点

1. 网站导航概述

ASP.NET 站点导航技术为用户导航站点提供一致的方法。ASP.NET 站点导航技术为站点创建统一、更容易管理的导航解决方案。

ASP.NET 站点导航提供如下功能：

✧ 站点地图：可以使用站点地图描述站点的逻辑结构，也可通过在添加或移除页面时修改站点地图（而不是修改所有网页的超链接）来管理页导航；

✧ ASP.NET 控件：可以使用 ASP.NET 控件在网页上显示导航菜单，导航菜单以站点地图为基础；

✧ 编程控件：可以以代码方式使用 ASP.NET 站点导航，以创建自定义导航控件或修改在导航菜单中显示的信息的位置；

✧ 访问规则：可以配置用于在导航菜单中显示或隐藏链接的访问规则；

✧ 自定义站点地图提供程序：可以创建自定义站点地图提供程序，以便使用自己的站点地图后端，并将提供程序插入 ASP.NET 站点导航系统。

2. ASP.NET 站点地图

要使用 ASP.NET 站点导航，必须描述站点结构以便站点导航 API 和站点导航控件可以正确公开站点结构。默认情况下，站点导航系统使用一个包含站点层次结构的 XML 文件。

创建站点地图最简单的方法是创建一个名为 Web.sitemap 的 XML 文件，该文件按站点的分层形式组织页面。ASP.NET 的默认站点地图提供程序自动选取此站点地图。

如代码清单 5-9 所示演示了站点地图如何查找一个三层结构的简单站点。url 属性可以以

快捷方式“~/”开头，该快捷方式表示应用程序根目录。

代码清单 5-9 站点地图查找三层结构的简单站点代码

```
<siteMap>
  <siteMapNode title="Home" description="Home" url="~/default.aspx">
    <siteMapNode title="Products" description="Our products"
      url="~/Products.aspx">
      <siteMapNode title="Hardware" description="Hardware choices"
        url="~/Hardware.aspx" />
      <siteMapNode title="Software" description="Software choices"
        url="~/Software.aspx" />
    </siteMapNode>
    <siteMapNode title="Services" description="Services we offer"
       url="~/Services.aspx">
       <siteMapNode title="Training" description="Training classes"
         url="~/Training.aspx" />
       <siteMapNode title="Consulting" description="Consulting services"
         url="~/Consulting.aspx" />
       <siteMapNode title="Support" description="Supports plans"
         url="~/Support.aspx" />
    </siteMapNode>
  </siteMapNode>
</siteMap>
```

在 Web.siteMap 文件中，为网站中的每一页添加一个 siteMapNode 元素，可以通过嵌入 siteMapNode 元素创建层次结构。在上例中，“硬件”和“软件”页是“产品”siteMapNode 元素的子元素。title 属性定义通常用作链接文本的文本，description 属性同时用作文档和 SiteMapPath 控件中的工具提示。

3. SiteMapPath 控件

SiteMapPath 控件用来显示一组文本或图像超链接，使访问者可以在使用最少页面空间的同时更轻松地定位网站。该控件是一种站点导航控件，反映 SiteMap 对象提供的数据。它提供了一种用于轻松定位站点的节省空间方式，用作当前显示页在站点中位置的引用点。

使用 SiteMapPath 控件后，页面将会在 SiteMapPath 控件的位置显示类似于“主页 > 文章管理 > 文章分类添加”这样的导航链接。SiteMapPath 控件直接使用网站的站点地图数据。需要注意的是如果将其用在没有包含在站点地图中的 Web 页时，则不会显示任何信息。

SiteMapPath 由节点组成。路径中的每个元素均称为节点，用 SiteMapNodeItem 对象表示。如表 5-19 所示描述了三种不同的节点类型。

表 5-19 SiteMapPath 控件的节点类型

节 点 类 型	说 明
根节点	锚定节点分层组的节点
父节点	有一个或多个子节点但不是当前节点的节点
当前节点	表示当前显示页的节点

SiteMapPath 控制显示的每个节点都是 HyperLink 或 Literal 控件，在模板或样式中会用到这两种控件。

SiteMapPath 控件使用的标准代码如下：

```
<asp:SiteMapPath ID="SiteMapPath1" runat="server"  />
```

其常用属性如下：

✧ CurrentNodeTemplate 属性：获取或设置一个控件模板，用于表示当前显示页的站点导航路径的节点；

✧ NodeStyle 属性：获取用于站点导航路径中所有节点的显示文本的样式；

✧ NodeTemplate 属性：获取或设置一个控件模板，用于站点导航路径的所有功能节点；

✧ PathDirection 属性：获取或设置导航路径节点的呈现顺序；

✧ PathSeperator 属性：获取或设置一个字符串，该字符串在呈现的导航路径中分隔 SiteMapPath 节点；

✧ PathSeperatorTemplate 属性：获取或设置一个控件模板，用于站点导航路径的路径分隔符；

✧ RootNodeTemplate 属性：获取或设置一个控件模板，用于站点导航路径的根节点；

✧ SiteMapProvider 属性：获取或设置用于呈现站点导航控件的 SiteMapProvider 的名称。

4. TreeView 控件

TreeView 控件用于在树结构中显示分层数据，如目录或文件目录，并且支持如下功能：

✧ 数据绑定，它允许控件的节点绑定到 XML、表格或关系数据；

✧ 站点导航，通过与 SiteMapDataSource 控件集成实现；

✧ 节点文本既可以显示为纯文本也可以显示为超链接；

✧ 借助编程方式访问 TreeView 对象模型，以动态地创建树、填充节点、设置属性等；

✧ 客户端节点填充（在支持的浏览器上）；

✧ 在每个节点旁显示复选框的功能；

✧ 通过主题、用户定义的图像和样式可实现自定义外观。

TreeView 控件由节点组成。树中的每个项都称为一个节点，它由 TreeNode 对象表示。节点类型的定义如下：

✧ 包含其他节点的节点称为父节点；

✧ 被其他节点包含的节点称为子节点；

✧ 没有子节点的节点称为叶节点；

✧ 不被其他任何节点包含，同时是所有其他节点的上级的节点是根节点。

一个节点可以同时是父节点和子节点，但是不能同时为根节点、父节点和叶节点。节点为根节点、父节点还是叶节点决定着节点的几种可视化属性和行为属性。节点可以处于以下两种状态之一，即选定状态和导航状态。默认情况下，会有一个节点处于选定状态。若要使一个节点处于导航状态，请将该节点的 NavigateUrl 属性值设置为空字符串以外的值；若要使一个节点处于选定状态，只要将该节点的 NavigateUrl 属性值设置为空字符串。

TreeView 控件的常用外观属性如下：

✧ CollapseImageToolTip 属性：获取或设置可折叠节点的指示符所显示图像的工具提示；

✧ CollapaseImageUrl 属性：获取或设置自定义图像的 URL，该图像用作可折叠节点的指示符；

✧ ExpandImageToolTip 属性：获取或设置可展开节点的指示符所显示图像的工具提示；

✧ ExpandImageUrl 属性：获取或设置自定义图像的 URL，该图像用作可展开节点的指示符；

✧ ImageSet 属性：获取或设置用于 TreeView 控件的图像组；

✧ LineImageFolder 属性：获取或设置文件夹的路径，该文件夹包含用于连接子节点和父节点的线条图像；

✧ NoExpandImageUrl 属性：获取或设置自定义图像的 URL，该图像用作不可展开节点的指示符；

✧ ShowExpandCollapse 属性：获取或设置一个值，它指示是否显示展开节点指示符；

✧ ShowLines 属性：获取或设置一个值，它指示是否显示连接子节点和父节点的线条；

✧ NodeIdent 属性：获取或设置 TreeView 控件子节点的缩进量（以像素为单位）；

✧ NodeWrap 属性：获取或设置一个值，它指示空间不足时节点中的文本是否换行。

TreeView 控件的常见行为属性如下；

✧ EnableClientScript 属性：获取或设置一个值，指示 TreeView 控件是否呈现客户端脚本以处理展开和折叠事件；

✧ ExpandDepth 属性：获取或设置第一次显示 TreeView 控件时所展开的层次数；

✧ MaxDataBindDepth 属性：获取或设置要绑定到 TreeView 控件的最大树级别数；

✧ PopulateNodesFromClient 属性：获取或设置一个值，它指示是否按需从客户端填充节点数据；

✧ ShowCheckBoxes 属性：获取或设置一个值，它指示哪些节点类型将在 TreeView 控件中显示复选框。

TreeView 控件的常用方法如下：

✧ CollapseAll()方法：关闭树中的每个节点；

✧ ExpandAll() 方法：打开树中的每个节点；

✧ FindNode(string)方法：检索 TreeView 控件中指定值路径处的 TreeNode 对象。

TreeView 控件提供多个可以对其进行编程的事件。TreeView 控件的常见事件如下；

✧ SelectedNodeChanged 事件：当选择 TreeView 控件中的节点时发生；

✧ TreeNodeExpanded 事件：当扩展 TreeView 控件中的节点时发生；

✧ TreeNodeCollapsed 事件：当折叠 TreeView 控件中的节点时发生。

下面的操作是在 TreeView 控件中显示分层数据的示例应用。

（1）创建数据的 XML 文件 BookStore.xml，该 XML 文件包含有关的书籍信息。 完成后的代码如代码清单 5-10 所示。

代码清单 5-10 BookStore.xml 源代码

```
<?xml version="1.0" standalone="yes"?>
<bookstore>
        <book ISBN="10-000000-001">
           <title>The Iliad and The Odyssey</title>
           <price>12.95</price>
        </book>
       <book ISBN="10-000000-999">
           <title>Anthology of World Literature</title>
           <price>24.95</price>
       </book>
</bookstore>
```

（2）新建 testtreeview.aspx 页，然后切换到“设计”视图。

（3）从“导航”组中将 TreeView 控件拖到页面上。

（4）选中 TreeView 控件，然后单击“显示智能标记”按钮。

（5）在 TreeView 任务菜单中的“选择数据源”下拉列表中选择“新建数据源”，出现“数据源配置向导”。

（6）在“应用程序从哪里获取数据”窗口中，选择“XML 文件”，对数据源保留默认 ID，单击“确定”按钮。

（7）在“配置数据源”对话框中，在“数据文件”框中输入“~/Bookstore.xml”，然后单击“确定”按钮。

（8）按【Ctrl+F5】组合键运行该页面。TreeView 控件的运行结果如图 5-9 所示。

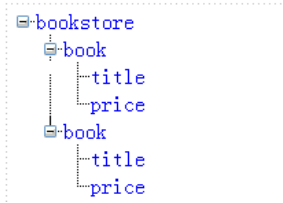

图 5-9　TreeView 控件的运行结果

5. Menu 控件

Menu 控件用于显示 Web 窗体页中的菜单，并常与用于导航网站的 SiteMapDataSource 控件结合使用。Menu 控件的常用功能如下：

✧ 数据绑定，将控件菜单项绑定到分层数据源；

✧ 站点导航，通过与 SiteMapDataSource 控件集成实现；

✧ 对 Menu 对象模型的编程访问，可动态创建菜单、填充菜单项、设置属性等；

✧ 可自定义外观，通过主题、用户定义图像、样式和用户定义模板实现；

✧ 用户单击菜单项时，Menu 控件可以导航到所链接的网页或直接回发到服务器。如果设置了菜单项的 NavigateUrl 属性，则 Menu 控件导航到所链接的页；否则，该控件将页回发到服务器进行处理。

Menu 控件显示两种类型的菜单，即静态菜单和动态菜单。静态菜单始终显示在 Menu 控件中。默认情况下，根级（级别 0）菜单项显示在静态菜单中。通过设置 StaticDisplayLevels 属性，可以在静态菜单中显示更多菜单级别（静态子菜单），级别高于 StaticDisplayLevels 属性所指定值的菜单项（如果有）显示在动态菜单中。仅当用户将鼠标指针置于包含动态子菜单的父菜单项上时，才会显示动态菜单。指定的持续时间过后，动态菜单将自动消失。使用 DisappearAfter 属性指定持续时间，还可以通过设置 MaximumDynamicDisplayLevels 属性，限制动态菜单的显示级别数，高于指定值的菜单级别则被丢弃。

Menu 控件由菜单项（由 MenuItem 对象表示）树组成。顶级菜单项称为根菜单项；具有父菜单项的菜单项称为子菜单项。所有根菜单项都存储在 Items 集合中，子菜单项存储在父菜单项的 ChildItems 集合中。

每个菜单项都具有 Text 属性和 Value 属性。Text 属性的值显示在 Menu 控件中，而 Value 属性则用于存储菜单项的任何其他数据（如传递给与菜单项关联的回发事件的数据）。在单击菜单项时，可导航到 NavigateUrl 属性指示的另一个网页。通过设置 ImageUrl 属性，也可选择在菜单项中显示图像。

Menu 控件可以使用任意分层数据源控件，如 XmlDataSource 控件或 SiteMapDataSource 控件。绑定到分层数据源控件时，只要将 Menu 控件的 DataSourceID 属性设置为数据源控件的 ID 值即可。Menu 控件自动绑定到指定的数据源控件，这是绑定到数据的首选方法。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 3

拓展任务号	5-3	任务名称	网站后台导航设计
计划用时	20 分钟	任务性质	课外
任务描述与目标			
根据网站结构分析，创建 ASP.NET 的站点地图，在后台母版页中添加 TreeView 控件和 SiteMapPath 控件，用于在后台不同页面间的快速导航			
主要操作步骤提示			
1．添加站点地图文件； 2．在母版页中添加 TreeView 控件，并进行相应的设置，用来显示导航目录； 3．在母版页中添加 SiteMapPath 控件，并进行相应的设置，用来显示当前访问者在网站中的位置； 4．测试后台所有页面，保证能在站内顺畅导航			

5.5 任务 4 非连接环境下的数据访问

5.5.1 DataSet 数据访问方式

在网络通信录项目中，使用 DataReader 类和 ADO.NET 提供的其他几个类，实现了数据的查询操作。但由于采用 DataReader 连接数据库时是面向连接的，读表时，只能向前读的，读完数据后由用户来决定是否要断开连接。而 DatSet 连接数据库时是非面向连接的，把表全部读到 SQL 的缓冲池中，断开数据库的连接，同时可以通过其他非连接对象对存储在缓冲池中的数据任意进行读取。显然后者更适用于 Web 项目。下面通过自定义通用的数据访问类 SqlDataAccess，学习 DataSet 类的基本用法。

一、实战演练

（1）在“解决方案资源管理器”中，用鼠标右键单击项目，创建一个类文件 SqlDbAccess.cs，Visual Studio 2012 开发环境会把新添加的类文件自动放入 App_Code 目录下。

（2）双击打开类文件，在类文件中，添加实现数据查询操作的方法代码。SqlDbAccess 类的部分代码如代码清单 5-11 所示。

代码清单 5-11 SqlDbAccess 类的部分代码

```
// Sql 数据库访问
 public class SqlDbAccess
 {
    private SqlConnection scon = null;
    private string constr = string.Empty;
    private static readonly SqlDbAccess model = new SqlDbAccess();
    public static SqlDbAccess GetDbModel(string constr)
    {
        model.constr = constr;
        return model;
     }
          public DataSet GetDataSet(string lpCmd) // 获取数据库数据
    {
        return GetDataSet(lpCmd, null);
    }
```

```
        public DataSet GetDataSet(string lpCmd, SqlParameter[] sqlparams)   //
获取数据库数据
        {
            return GetDataSet(lpCmd, CommandType.Text, sqlparams);
        }
        // 获取数据库数据
        public  DataSet  GetDataSet(string  lpCmd,  CommandType  cmdtype,
SqlParameter[] sqlparams)
        {
            this.Open();
            SqlCommand sqlcom = new SqlCommand(lpCmd, this.scon);
            sqlcom.CommandType = cmdtype;
            if (sqlparams != null && sqlparams.Length > 0)
            {
                foreach (SqlParameter sp in sqlparams)
                    sqlcom.Parameters.Add(sp);
            }
            DataSet ds = new DataSet();
            SqlDataAdapter adpter = new SqlDataAdapter(sqlcom);
            adpter.Fill(ds);
            this.Close();
            return ds;
        }
    }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-20 所示。

表 5-20　演练完成情况评价表

任务号	5-10	任务名称	通用数据库操作类 SqlHelper 的使用
任务子项	完成情况	主要问题	未完成原因
创建 SqlHelper 类文件			
添加类方法			

三、知识点

在项目 2 中介绍的以连接方式访问数据库，只允许将数据作为仅向前型的只读数据流进行查看。如果要对查询结果进行排序、搜索、筛选等操作，应当怎么做呢？ADO.NET 对象模型还提供了这种功能的类。这些类可以作为脱机数据缓存，将查询结果导入 DataTable 之后，就可以关闭与数据源的连接，并继续使用数据。这种非连接类涉及多个对象。

1. DataSet 类

DataSet 类是 ADO.NET 结构的主要组件。DataSet 对象包含一个数据集，由一组 DataTable 对象组成，通过 DataRelation 类可以关联这些对象，可以将 DataSet 对象看成许多 DataTable 对象的容器。借助 UniqueConstraint 和 ForeignKeyConstraint 对象可保证 DataSet 中数据的完整性。在 ADO.NET 的模型中，可以把 DataSet 看成一个数据库的缩影。

DataSet 类可将数据和架构作为 XML 文档进行读/写。数据和架构可通过 HTTP 传输，并在支持 XML 的任何平台上被任何应用程序使用。可使用 WriteXmlSchema 方法将架构保存为 XML 架构，并且可以使用 WriteXml 方法保存架构和数据，若要读取既包含架构也包含数

据的 XML 文档，请使用 ReadXml 方法。

DataSet 类是一个非常丰富而完整的内存中的数据容器，它反映了一个 DBMS 的结构和某些功能。也就是说，DataSet 类是内存中的数据容器，它不仅可以存储数据，而且可以存储数据的关系结构，并且可执行各种有用的操作，如排序和过滤数据、从 XML 填充它自己或者把数据和模式导出到 XML。

DataSet 对象可以通过如下方法从数据源获取数据并填充：

✧ 调用 DataAdapter 对象的 Fill 方法；

✧ 通过程序创建 DataRow 对象，给 DataRow 对象的各列进行赋值，然后把 DataRow 对象添加到 Rows 集合中；

✧ 将 XML 文档或流读入到 DataSet 对象中。

DataSet 类的基本创建方法如下：

```
DataSet ds = new DataSet();
SqlDataAdapter adpter = new SqlDataAdapter(sqlcom);
adpter.Fill(ds);
```

DataSet 类的常用属性和方法如下：

✧ DataSetName 属性：获取或设置当前 DataSet 的名称；

✧ Tables 属性：获取包含在 DataSet 中表的集合；

✧ Clear()方法：通过移除所有表中的所有行来清除任何数据的 DataSet；

✧ Clone()方法：复制 DataSet 的结构，包括所有 DataTable 架构、关系和约束，不复制任何数据；

✧ Copy()方法：复制该 DataSet 的结构和数据；

✧ GetXml()方法：返回存储在 DataSet 中数据的 XML 表示形式；

✧ GetXmlSchema()方法：返回存储在 DataSet 中数据的 XML 表示形式的 XML 架构；

✧ ReadXml()方法：将 XML 架构和数据读入 DataSet；

✧ ReadXmlSchema()方法：将 XML 架构读入 DataSet；

✧ WriteXml()方法：从 DataSet 写 XML 数据，还可以选择写架构；

✧ WriteXmlSchema()方法：写 XML 架构形式的 DataSet 结构。

2. DataTable 类

DataTable 类是 ADO.NET 库中的核心对象。DataTable 类允许通过行和列的集合查看对象，可以通过 DataAdapter 对象的 Fill 方法将查询结果存储到 DataTable 类中。基本代码如下：

```
DataTable  dtbl = new DataTable ();
SqlDataAdapter adpter = new SqlDataAdapter(sqlcom);
adpter.Fill(dtbl);
```

DataTable 类包含了其他非连接对象的集合。DataTable 类的 Rows 属性会返回一个 DataRow 对象的集合，通过这个集合可以查看表中的所有内容，通过 Columns 属性可以获取 DataColumns 对象的集合，从而可以查看 DataTable 的结构。

DataTable 类的常用属性如下：

✧ Columns 属性：获取属于该表的列的集合；

✧ Rows 属性：获取属于该表的行的集合；

✧ TableName 属性：获取或设置 DataTable 的名称。

其常用方法与 DataSet 类类似，这里不再叙述。

3. DataRow 类

DataRow 和 DataColumn 对象是 DataTable 类的主要组件，使用 DataRow 对象及其属性和方法检索、评估、插入、删除和更新 DataTable 类中的值。DataRow 类的常见用法如下：

```
DataRow row = dtbl.NewRow();
    row["fName"] = "John";
    row["lName"] = "Smith";
dtbl.Rows.Add(row);
```

DataRow 类的常用属性和方法如下：

✧ Item 属性：获取或设置存储在指定列中的数据；

✧ Delete()方法：删除 DataRow 类。

4. DataColumn 类

用于创建 DataTable 类架构的基本构造块。一个 DataColumn 对象对应 DataTable 类中的一列。事实上 DataColumn 对象并非包含存储在 DataTable 类中的数据，而是存储有关该列的信息。DataColumn 类的常见用法如下：

```
DataColumn column = new DataColumn();
column.DataType = System.Type.GetType("System.Decimal");
column.AllowDBNull = false;
column.Caption = "Price";
column.ColumnName = "Price";
column.DefaultValue = 25;
dtbl.Columns.Add(column);
DataRow row;
for(int i = 0; i < 10; i++)
{
    row = dtbl.NewRow();
    row["Price"] = i + 1;
    table.Rows.Add(row);
}
```

5. DataView 类

表示用于排序、筛选、搜索、编辑和导航的 DataTable 类的可绑定数据的自定义视图。在将一个查询结果置于 DataTable 对象中之后，就可以用 DataView 对象以不同的方式查看数据。

DataView 对象与数据库视图类似，DataView 类提供了可向其应用不同排序和筛选条件的单个数据集的动态视图。但是，与数据库视图不同的是，DataView 类不能作为表来处理，无法提供连接表的视图，另外，还不能排除存在于源表中的列，也不能追加不存在于源表中的列。

一个 DataTable 类可以使用多个 DataView 对象同时进行查看，如在窗体中可以拥有两个表格，一个按字母排序显示所有用户，一个按另一个字段排序显示。

使用 DataView 构造函数，也可以创建对 DataTable 类的 DefaultView 属性的引用。DataView 类的常见用法如下：

```
DataView view = new DataView(dtbl);
DataView view = dtbl.DefaultView;
```

DataView 类的常用属性和方法如下：

✧ AllowDelete 属性：设置或获取指示 DataView 对象是否允许删除；

✧ AllowEdit 属性：获取或设置指示 DataView 对象是否允许编辑；

✧ AllowNew 属性：获取或设置指示 DataView 对象是否可以使用 AddNew 方法添加新行；

✧ Count 属性：获取 DataView 类中记录的数量；

✧ Item 属性：从指定的表获取一行数据；

✧ Sort 属性：获取或设置 DataView 类的一个或多个排序列及排序顺序；

✧ Table 属性：获取或设置源 DataTable 类；

✧ AddNew()方法：将新行添加到 DataView 类中；

✧ Delete()方法：删除指定索引位置的行。

✧ Find()方法：按指定的排序关键字值在 DataView 中查找行；

✧ FindRows()方法：返回 DataRowView 对象的数组，这些对象的列与指定的排序关键字值相匹配。

在非连接方式的数据操作管理中，还有许多相关的类：

✧ DataColumnCollection 类：用于表示 DataTable 类的 DataColumn 对象的集合；

✧ DataRelation 类：用于表示两个 DataTable 对象之间的父子关系；

✧ DataRelationCollection 类：用于表示此 DataSet 类的 DataRelation 对象的集合；

✧ DataRowCollection 类：用于表示 DataTable 类的行的集合；

✧ DataTableCollection 类：用于表示 DataSet 类的表的集合。

6. DataAdapter 类

DataAdapter 类负责把数据库的数据填充到非连接的 DataSet 类。DataAdapter 类根据不同的数据源提供不同的程序。SqlDataAdapter 是 DataSet 类和 SQL Server 之间的桥接器，用于填充 DataSet 类和更新数据源。SqlDataAdapter 通过对数据源使用适当的 SQL 语句映射 Fill 和 Update 来提供这一桥接。

SqlDataAdapter 常与 SqlConnection 和 SqlCommand 一起使用，以便在连接到 SQL Server 数据库时提高性能。如下代码演示了 SqlDataAdapter 对象常见的方式：

```
SqlConnection connection =new SqlConnection(connectionString))
SqlDataAdapter adapter = new SqlDataAdapter();
adapter.SelectCommand = new SqlCommand(queryString, connection);
adapter.Fill(dataset);
```

DataAdapter 对象的常用属性和方法如下：

✧ SelectCommand 属性：获取或设置用于在数据源中选择记录的命令；

✧ InsertCommand 属性：获取或设置用于将新记录插入到数据源中的命令；

✧ UpdateCommand 属性：获取或设置用于更新数据源中记录的命令；

✧ DeleteCommand 属性：获取或设置用于从数据集中删除记录的命令；

✧ Fill()方法：从数据源中提取数据以填充数据集；

✧ Update()方法：更新数据源。

四、任务拓展

本节完成 5 个拓展实践任务，2 个课内拓展实践任务和 3 个课外拓展实践任务。

拓展任务卡 4

拓展任务号	5-4	任务名称	在类库 Model 中创建新闻及新闻类别业务实体类
计划用时	20 分钟	任务性质	课内

任务描述与目标

企业网站项目采用多层模式设计思想，整个项目解决方案由 5 个项目组成，分别为网站 Web、类库 BLL、类库 DAL、类库 Model 和类库 Common。其中网站 Web 对应表示层，主要包括网页页面、用户控件等，类库 BLL 中包含所有业务逻辑层中的类，类库 DAL 中包含所有数据访问层中的类，类库 Model 中则包含所有业务实体类，类库 Common 包含公共类。

在这个任务中，主要完成 Web 外的其他类库的创建和在类库 Model 中创建新闻及新闻类别业务实体类操作

主要操作步骤提示

1．在解决方案中，添加名称分别为 BLL、DAL、Model、Common 的四个类库；

2．在类库 Model 中创建新闻类别实体类 ArticleClass，参考代码如下：

```
public class ArticleClass
   {
        #region Public Properties
        [DataObjectFieldAttribute(true, true, false)]
        public int ac_id { get; set; }//类别 ID
        public string ac_name { get; set; }    //新闻类别名称
        public int parent_id { get; set; }    //父类 ID
        public int ac_order { get; set; }    //类别顺序
        #endregion
    }
```

3．在类库 Model 中创建新闻实体类 Article，代码如下：

```
public class Article{}
```

4．在类库 Model 中创建所有实体集合类的基类 CollectionBase，参考代码如下：

```
public class CollectionBase<T> : Collection<T>
    {
        public CollectionBase() : base(new List<T>()) { }
        public CollectionBase(IList<T> initialList) : base(initialList) { }
    }
```

5．在类库 Model 中创建新闻集合类 ArticleCollection，代码如下：

```
public class ArticleCollection : CollectionBase<Article> { }
```

6．在类库 Model 中创建新闻类别集合类，代码如下：

```
public class ArticleClassCollection : CollectionBase<ArticleClass> { }
```

拓展任务卡 5

拓展任务号	5-5	任务名称	在类库 Model 中创建项目的其他业务实体类
计划用时	60 分钟	任务性质	课外

任务描述与目标

在这个任务中，主要完成类库 Model 中除新闻及新闻类别业务实体类之外的其他实体类的创建工作

主要操作步骤提示

1．结合数据库，分析和设计其他实体类；

2．具体要开发的实体类如下所示：

网站公告实体类：Note；网站设置信息实体类：SiteProperty；

产品实体类：Product；产品类别实体类：ProductClass；

续表

拓展任务号	5-5	任务名称	在类库 Model 中创建项目的其他业务实体类
计划用时	60 分钟	任务性质	课外
留言实体类：Feedback；管理员实体类：Admin； 留言集合类：FeedbackCollection；产品集合类：ProductCollection； 管理员集合类：AdminCollection；产品类别集合类：ProductClassCollection。			

拓展任务卡 6

拓展任务号	5-6	任务名称	在类库 DAL 中创建新闻类别数据访问类
计划用时	20 分钟	任务性质	课内
任务描述与目标			
在这个任务中，主要完成在类库 DAL 中创建新闻类别数据访问类的操作，该类实现对新闻类别表中数据的 CRUD 基本操作			
主要操作步骤提示			

1．在类库 DAL 中创建 AppConfiguration 类，用于读取 Web.config 中配置的数据加连接字符串，参考代码如下：

```
public static class AppConfiguration
  { public static string ConnectionString
    { get {
       return
ConfigurationManager.ConnectionStrings["EnterpriseWebConnectionString"].ConnectionString;
      }
    }
    }
```

2．在类库 DAL 中创建 ArticleClassDal 类，实现新闻类别数据的操作，参考方法及主要代码如下：

```
public static ArticleClassCollection GetList()    //查询文章分类所有数据
public static ArticleClassCollection GetListByParentID(int parent_id)    //条件查询文章分类
public static ArticleClass GetItem(int ac_id)    //获得文章分类
public static bool Delete(int ac_id)    //删除一条文章分类
public static bool Update(ArticleClass myArticleClass)    //修改文章分类
public static bool Insert(ArticleClass myArticleClass)    //插入文章分类
  //获得文章分类树记录
public static DataSet GetTree()
  {
    DataSet ds = new DataSet();
    using (   SqlConnection myConnection=
           new SqlConnection(AppConfiguration.ConnectionString))
    {
      using (SqlDataAdapter myAdapter = new SqlDataAdapter())
      {
      SqlCommand myCommand=
        new SqlCommand("sprocArticleClassGetTree", myConnection);
      myCommand.CommandType= CommandType.StoredProcedure;
            myAdapter.SelectCommand = myCommand;
            myAdapter.Fill(ds, "ArticleClass");
        }
    }
  return ds;
}
  //初始化一个 ArticleClass 类实体，并填充数据
private static ArticleClass FillDataRecord(IDataRecord myDataRecord)
{
  ArticleClass myArticleClass = new ArticleClass();
```

续表

拓展任务号	5-6	任务名称	在类库 DAL 中创建新闻类别数据访问类
计划用时	20 分钟	任务性质	课内

```
  myArticleClass.ac_id= myDataRecord.GetInt32(myDataRecord.GetOrdinal("ac_id"));
  myArticleClass.ac_name= myDataRecord.GetString(myDataRecord.GetOrdinal("ac_name"));
  if (!myDataRecord.IsDBNull(myDataRecord.GetOrdinal("parent_id")))
  {
    myArticleClass.parent_id=
        myDataRecord.GetInt32(myDataRecord.GetOrdinal("parent_id"));
  }
  myArticleClass.ac_order= myDataRecord.GetInt32(myDataRecord.GetOrdinal("ac_order"));
  return myArticleClass;
}
```

拓展任务卡 7

拓展任务号	5-7	任务名称	在类库 DAL 中创建项目的其他数据访问类
计划用时	120 分钟	任务性质	课外

任务描述与目标

在这个任务中，主要完成类库 DAL 中除新闻类别 DAL 类之外的其他数据访问类的创建工作

主要操作步骤提示

1．参考新闻分类 DAL，分析和设计其他 DAL 类；

2．具体要开发的 DAL 类如下所示：

管理员 DAL 类：AdminDal

新闻 DAL 类：ArticleDal

留言 DAL 类：FeedbackDal

产品 DAL 类：ProductDal

产品类别 DAL 类：ProductClassDal

站点信息 DAL 类：SitePropertyDal

拓展任务卡 8

拓展任务号	5-8	任务名称	在类库 BLL 中创建项目的其他数据访问类
计划用时	80 分钟	任务性质	课外

任务描述与目标

在这个任务中，主要完成类库 BLL 中除新闻类别 BLL 类之外的其他业务逻辑类的创建工作

主要操作步骤提示

1．参考新闻分类 BLL，分析和设计其他 BLL 类；

2．具体要开发的 BLL 类如下所示：

管理员 BLL 类：AdminBll

新闻 BLL 类：ArticleBll

留言 BLL 类：FeedbackBll

产品 BLL 类：ProductBll

产品类别 BLL 类：ProductClassBll

站点信息 BLL 类：SitePropertyBll

5.5.2 数据绑定控件

几乎所有的 Web 应用程序都要和数据打交道，无论这些数据采用哪种存储方式（数据库或者 XML 文件或者其他方式）。读取这些数据时，都需要一个方便灵活且有吸引力的方式把数据显示到网页上。ASP.NET 为我们提供了一个丰富全能的数据绑定模型。下面以企业网站首页制作为例，介绍数据绑定及数据绑定控件的相关知识。

一、实战演练

（1）在“解决方案资源管理器”中选中企业网站 Web 项目，添加“Web 窗体”，“母版页”选择“Web.master”，窗体的名称设为“index.aspx”。

（2）在“设计”视图中添加控件，网站首页设计效果如图 5-10 所示。

图 5-10 网站首页设计效果

（3）首页设计过程中涉及的控件及控件属性如表 5-21 所示。

表 5-21 首页设计过程中涉及的控件及控件属性

控 件	ID	其 他 属 性
产品列表显示控件 DataList	DataList1	DataKeyField="prod_id" DataSourceID="ObjectDataSourceProduct" GridLines="Horizontal"
新闻列表控件 Repeater	Repeater1	dataSourceID="ObjectDataSourceArticle1"
产品列表的数据源控件 ObjectDataSource	ObjectDataSourceProduct	OldValuesParameterFormatString="original_{0}" SelectMethod="GetList" TypeName="EnterpriseWeb.Bll.ProductBll"
新闻列表的数据源控件 ObjectDataSource	ObjectDataSourceArticle1	OldValuesParameterFormatString="original_{0}"SelectMethod="GetList" TypeName="EnterpriseWeb.Bll.ArticleBll"

（4）index.aspx 页面的主要代码如代码清单 5-12 所示。

代码清单 5-12　index.aspx 页面的主要代码

```
    <asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="
Server">
     <div class="innerContent">
       <span class="titleContent">首页 </span>
          <div class="info" >
           <table>
            <thead><tr><th>公司介绍</th></tr></thead>
            <tr><td class="artContent"></td></tr>
           </table>
            <asp:DataList ID="DataList1" runat="server" DataKeyField="prod_id"
              DataSourceID="ObjectDataSourceProduct" RepeatColumns="3"
              GridLines="Horizontal">
                <HeaderTemplate> 产品</HeaderTemplate>
                <ItemTemplate>
                <table style="width:90%;">
                  <tr><td class="noborder center">
                  <a href='<%# "ProductShow.aspx?id=" + Eval("prod_id") %>'>
                  <asp:Image  ID="imgProd"   runat="server"  ImageUrl='<%#  Eval("
prod_image") %>'
                   AlternateText='<%# Eval("prod_name") %>' CssClass="imgProduct"
/></a>
                   </td></tr>
                   <tr><td class="noborder center" >
                     <asp:HyperLink ID="HyperLink2" runat="server"
                       NavigateUrl='<%#          "ProductShow.aspx?id="          +
Eval("prod_id") %>'
                        Text='<%# Eval("prod_name")%>'
                        CssClass="LabelStyle"></asp:HyperLink>
                    </td></tr>
                 </table>
                </ItemTemplate>
                  <FooterTemplate></table></FooterTemplate>
                </asp:DataList>
            <asp:ObjectDataSource ID="ObjectDataSourceProduct" runat="server"

OldValuesParameterFormatString="original_{0}"SelectMethod="GetList"
                   TypeName="EnterpriseWeb.Bll.ProductBll">
              <SelectParameters>
                  <asp:Parameter DefaultValue="0" Name="pc_id" Type="Int32" />
                <asp:Parameter DefaultValue="1" Name="startRowIndex" Type="Int32" />
                  <asp:Parameter DefaultValue="3" Name="maximumRows" Type="Int32" />
             </SelectParameters>
           </asp:ObjectDataSource>
           <asp:Repeater ID="Repeater1" runat="server"
                  DataSourceID="ObjectDataSourceArticle1">
             <HeaderTemplate><table><thead>
                  <tr><th>新闻</th></tr></thead></HeaderTemplate>
              <ItemTemplate>
                  <tr><td class="noborder" >
                  <asp:HyperLink ID="HyperLink2" runat="server"
                    NavigateUrl='<%# "ArticleShow.aspx?id=" + Eval("art_id") %>'
                    Text='<%#Eval("art_title")%>'
                   CssClass="LabelStyle"></asp:HyperLink>  
```

```
                <%#string.Format("{0:d}", Eval("art_date")) %>
                </td></tr>
              </ItemTemplate>
               <FooterTemplate></table></FooterTemplate>
             </asp:Repeater>
            <asp:ObjectDataSource ID="ObjectDataSourceArticle1" runat="server"
                OldValuesParameterFormatString="original_{0}"
                SelectMethod="GetList"
TypeName="EnterpriseWeb.Bll.ArticleBll">
                <SelectParameters>
                 <asp:Parameter DefaultValue="2" Name="ac_id" Type="Int32" />
                <asp:Parameter DefaultValue="1" Name="startRowIndex" Type="Int32" />
                <asp:Parameter DefaultValue="10" Name="maximumRows" Type="Int32" />
                </SelectParameters>
             </asp:ObjectDataSource>
           </div>
      </div>
    </asp:Content>
```

（5）编写首页后台 Page_Load 事件响应代码。完成后的代码如代码清单 5-13 所示。

代码清单 5-13 首页后台 Page_Load 事件响应代码

```
    protected void Page_Load(object sender, EventArgs e)
        {
           if (!IsPostBack)
           {//获取网站的站点名，使首页的标题显示为“嘉兴嘉诚网络—首页”
              Master.SetSiteProperty();
              Page.Title = SitePropertyBll.GetSiteProperty().site_name + "—首
页";
            }
        }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-22 所示。

表 5-22 演练完成情况评价表

任务号	5-11	任务名称	前台首页制作
任务子项	完成情况	主要问题	未完成原因
首页的设计			
后台事件代码的开发			

三、知识点

1. 数据绑定

使用 ASP.NET 数据绑定，可以将任何服务器控件绑定到简单的属性、集合、表达式或方法中。在数据绑定中，数据的显示是由数据源和数据显示控件共同决定的。数据源决定数据的内容，数据显示控件决定数据的显示方式。数据绑定的作用机制本质上就是由数据显示控件调

用数据源的方法从而获得数据。

数据绑定的语法如下：

```
<语言标记 …属性='<%# 数据绑定表达式 %>' runat="server">
```

如果表达式的结果直接输出到网页上，那么数据绑定的语法如下：

```
字符串：<%# 数据绑定表达式 %>
```

该语法是在 .aspx 页中使用数据绑定的基础，所有数据绑定表达式都必须包含在这些字符中。常见的绑定方式如下。

简单属性

```
<%# custID %>
```

集合（用于订单的语法）

```
<asp:ListBox  id="List1"  datasource='<%# myArray %>'  runat="server">
```

表达式（用于联系人的语法）

```
<%# ( customer.First Name + " " + customer.LastName ) %>
```

方法结果（用于未结清余额的语法）

```
<%# GetBalance(custID) %>
```

使用 TextBox Web 服务器控件

```
<asp:textbox id=txt text="<%# custID %>" runat=server />
```

为.aspx 页上的对象确定并设置了特定数据源后，必须将数据绑定到这些数据源。可以使用 Page.DataBind 或 Control.DataBind 方法将数据绑定到数据源。这两种方法的使用方式很相似，主要差别在于调用 Page.DataBind 方法后，所有数据源都将绑定到它们的服务器控件。在显式调用 Web 服务器控件的 DataBind 方法或在调用页面级的 Page.DataBind()方法之前，不会有任何数据呈现给控件。通常，可以通过 Page_Load 事件调用 Page.DataBind()方法。

2. 数据绑定列表控件

数据绑定列表控件是可以绑定到集合的特殊的 Web 服务器控件，可以使用这些控件以自定义的模板格式显示数据行。所有列表控件都公开 DataSource 和 DataMember 属性，这些属性用于绑定到集合。ASP.NET 提供了丰富的数据绑定控件，根据这些绑定控件，可以很轻松地在页面中显示数据。

3. DataList 控件

DataList 控件以表的形式呈现数据，通过该控件，可以使用不同的布局来显示数据记录，如将数据记录排成列或行的形式。对 DataList 控件进行配置，还能够编辑或删除表中的记录。

DataList 服务器控件具有默认的布局和外观，通过使用模板可以自定义其外观。模板是一组 HTML 元素和控件，它们构成控件特定部分的布局，如在 DataList 控件中，可以使用 HTML 元素和控件的组合来创建列表中每行的布局。DataList 控件的格式请参见代码清单 5-14。

DataList 控件的常用属性如下：

✧ AltermatingItemStyle 属性：交替项的样式；

✧ EditItemStyle 属性：正在编辑项的样式；

✧ FooterStyle 属性：列表结尾处脚注的样式；

✧ HeaderStyle 属性：列表头部标头的样式；

✧ ItemStyle 属性：单个项的样式；

✧ SelectedItemStyle 属性：选定项的样式；

✧ SeparatorStyle 属性：各项之间分隔符的样式；

✧ RepeatDirection 属性：获取或设置 DataList 控件是垂直显示还是水平显示；

✧ RepeatLayout 属性：获取或设置控件是在表中显示还是在流布局中显示；

✧ RepeatColumns 属性：获取或设置要在 DataList 控件中显示的列数。

通过为 DataList 控件的不同部分设置样式属性可以自定义该控件的外观。借助 DataList 控件的自动套用格式设置也可以实现较多的外观效果，如图 5-11 所示。

利用属性生成器，同样可以通过选择相应的项目来生成属性，这些属性能够极大地方便开发人员制作 DataList 控件的界面样式，如图 5-12 所示。

对模板项中的按钮进行操作时，如果按钮的 CommandName 属性为 edit，则该按钮可以引发 EditCommand 事件，同样也可以配置不同的 CommandName 属性来实现不同的操作。

图 5-11 “自动套用格式”对话框

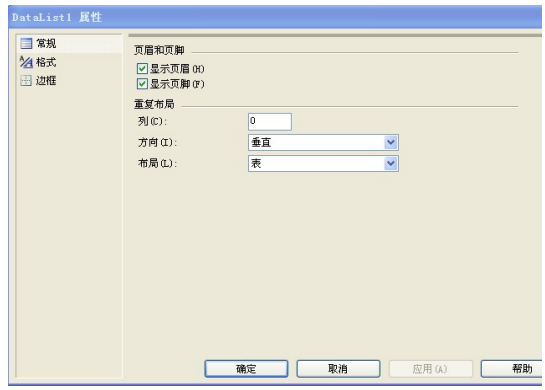

图 5-12 属性生成器

有关 DataList 事件处理的示例代码如代码清单 5-14 所示。

代码清单 5-14 DataList 事件处理的示例代码

```
    <asp:DataList ID="DataList1" runat="server" BackColor="White"
    BorderColor="#E7E7FF" BorderStyle="None" BorderWidth="1px" CellPadding="3"
    DataKeyField="ID" DataSourceID="SqlDataSource1" Font-Bold="False"
    Font-Italic="False" Font-Overline="False" Font-Strikeout="False"
    Font-Underline="False" GridLines="Horizontal" Width="100%"
    ondeletecommand="DataList1_DeleteCommand">
        <FooterStyle BackColor="#B5C7DE" ForeColor="#4A3C8C" />
        <AlternatingItemStyle BackColor="#F7F7F7" />
        <ItemStyle BackColor="#E7E7FF" ForeColor="#4A3C8C" />
       <SelectedItemStyle BackColor="#738A9C" Font-Bold="True" ForeColor="
#F7F7F7" />
        <HeaderStyle BackColor="#4A3C8C" Font-Bold="True" ForeColor="#F7F7F7" />
    <ItemTemplate>
        新闻 ID:
        <asp:Label ID="IDLabel" runat="server" Text='<%# Eval("ID") %>' />
        <br />
        新闻编号:
```

```
        <asp:Label ID="TITLELabel" runat="server" Text='<%# Eval("TITLE") %>' />
         <br />
         <asp:Button ID="Button1" runat="server" Text="删除"
         CommandName="delete" CommandArgument='<%# Eval("ID") %>'/>
      </ItemTemplate>
   </asp:DataList>
```

代码清单 5-14 中使用了数据绑定表达式语法，数据绑定表达式包含在<%#和%>分隔符之内，并使用了 Eval 和 Bind 函数。Eval 函数用于定义单向（只读）绑定，Bind 函数用于定义双向（可更新）绑定。

示例中创建了一个 DataList 控件并配置了按钮控件，并将按钮控件的 CommandName 属性配置为“delete”，触发该按钮则会引发 DeleteCommand 事件。在属性窗口中找到 DeleteCommand 事件，双击“DeleteCommand”连接系统会自动生成 DeleteCommand 事件相应的方法。当生成了 DeleteCommand 事件后，可以在代码段中编写相应的方法，示例代码如下：

```
    protected void DataList1_DeleteCommand(object source, DataListCommand
EventArgs e)
    {
          Label1.Text = e. CommandArgument.ToString()+"被执行";
    }
```

当用户单击了相应的按钮时会触发 DeleteCommand 事件。运行结果如图 5-13 所示。

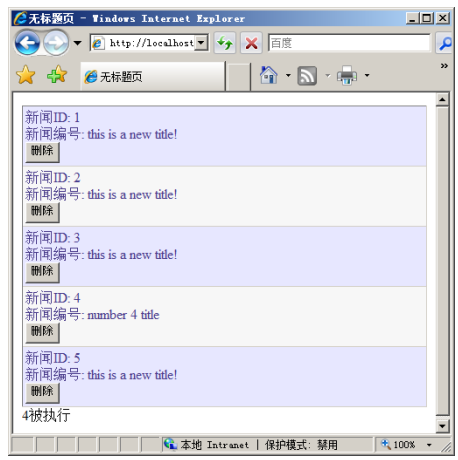

图 5-13　触发 DeleteCommand 事件运行结果

程序运行后，当用户单击了相应的按钮时，开发人员可以通过获取传递的 CommandArgument 参数的值来编写相应的方法从而实现不同的应用。

4. Repeater 控件

Repeater 控件是一个基本模板数据绑定列表。它没有内置的布局或样式，因此必须在该控件的模板内显式声明所有的布局、格式设置和样式标记。Repeater 控件的格式参见代码清单 5-15。

Repeater 控件允许在模板间拆分标记。 若要利用模板创建表，则在 HeaderTemplate 中包含表开始标记（<table>），在 ItemTemplate 中包含单个表行标记（<tr>），并在 FooterTemplate 中包含表结束标记（</table>）。

Repeater 控件没有内置的选择功能和编辑支持。可以使用 ItemCommand 事件来处理从模板引发到该控件的控件事件。

如表 5-23 所示描述了 Repeater 控件支持的模板。

表 5-23　Repeater 控件支持的模板

模 板 名	说　明
AlternatingItemTemplate	如果已定义，则为 DataList 中的交替项提供内容和布局；如果未定义，则使用 ItemTemplate
FooterTemplate	如果已定义，则为 DataList 的脚注部分提供内容和布局；如果未定义，将不显示脚注部分

续表

模 板 名	说 明
HeaderTemplate	如果已定义，则为 DataList 的页眉节提供内容和布局；如果未定义，将不显示页眉节
ItemTemplate	为 DataList 中的项提供内容和布局所要求的模板
SeparatorTemplate	如果已定义，则为 DataList 中各项之间的分隔符提供内容和布局；如果未定义，将不显示分隔符

如代码清单 5-15 所示演示了如何使用 Repeater 控件在 HTML 表中显示数据。

代码清单 5-15 使用 Repeater 控件在 HTML 表中显示数据的代码

```
<asp:Repeater ID="Repeater1" runat="server"  >
      <HeaderTemplate>
         <table><tr><th>Name</th><th>Description</th></tr>
      </HeaderTemplate>
      <ItemTemplate>
         <tr><td style="background-color:#CCFFCC">
          <asp:Label  runat="server"  ID="Label1"  Text='<%#  Eval("Category
Name") %>' />
          </td>
          <td style="background-color:#CCFFCC">
          <asp:Label       runat="server"       ID="Label2"       Text='<%#
Eval("Description") %>' />
          </td>
          </tr>
      </ItemTemplate>
      <AlternatingItemTemplate>
          <tr>
          <td> <asp:Label  runat="server"  ID="Label3"  Text='<%# Eval("Category
Name") %>' /> </td>
          <td>     <asp:Label     runat="server"     ID="Label4"     Text='<%#
Eval("Description") %>' /> </td>
          </tr>
      </AlternatingItemTemplate>
      <FooterTemplate>
        </table>
      </FooterTemplate>
</asp:Repeater>
```

5. 数据源控件

ASP.NET 包含一些数据源控件，这些数据源控件允许使用不同类型的数据源，如数据库、XML 文件或中间层业务对象。数据源控件连接到数据源，从中检索数据，并使得其他控件可以绑定到数据源而无需代码。数据源控件还支持修改数据。

.NET Framework 包含支持不同数据绑定方案的数据源控件。如表 5-24 所示描述了内置的数据源控件。

表 5-24 内置的数据源控件

数据源控件	说 明
LinqDataSource	使用此控件，可以通过标记在 ASP.NET 网页中使用语言集成查询 (LINQ)，从数据对象中检索和修改数据，支持自动生成选择、更新、插入和删除命令、还支持排序、筛选和分页
ObjectDataSource	使用业务对象或其他类，以及创建依赖中间层对象管理数据的 Web 应用程序，支持对其他数据源控件不可用的高级排序和分页方案

续表

数据源控件	说　明
SqlDataSource	使用 Microsoft SQL Server、OLE DB、ODBC 或 Oracle 数据库，与 SQL Server 一起使用时支持高级缓存功能。当数据作为 DataSet 对象返回时，此控件还支持排序、筛选和分页
AccessDataSource	使用 Microsoft Access 数据库。当数据作为 DataSet 对象返回时，支持排序、筛选和分页
SiteMapDataSource	ASP.NET 站点导航使用

使用数据源控件的方法很简单。要定义一个 AccessDataSource，可以向页面添加如下代码：

```
<asp:AccessDataSource ID="dsBooks" runat="server"
  DataFile="App_Data/BookCatalogSystem.mdb"
   SelectCommand="Select AuthorId,AuthorName from Authors" />
```

在这个例子中，DataFile 属性指定了数据库文件的位置，而 SelectCommand 属性指定了控件提供对哪种数据的访问。定义了数据源控件，只要把数据绑定控件的 DataSourceID 属性设为数据源控件的 ID 就可以显示数据了。如要在一个下拉列表中显示 Auhtors 表的 AuthorName 字段，可以使用如下代码：

```
<asp:DropDownList ID="drpAuthors" runat="server"
  DataSourceID="dsBooks" DataTextFiled="AuthorName" />
```

SqlDataSource 的使用和 AccessDataSource 的使用类似，这里就不再详细介绍。

虽然 SqlDataSource、AccessDataSource 数据源控件允许执行常见的数据获取任务时无需编程，但由于使用这些控件后数据访问细节就位于表示层中，不利于软件的分层复用，因而在实际开发项目中基本不会使用。

ObjectDataSource 控件表示具有数据检索和更新功能的中间层对象。作为数据绑定控件（如 GridView、FormView 或 DetailsView 控件）的数据接口，ObjectDataSource 控件可以使这些控件在 ASP.NET 网页上显示和编辑中间层业务对象中的数据。

ObjectDataSource 控件通过提供一种将相关页上的数据控件绑定到中间层业务对象的方法，为三层结构提供支持。在不使用扩展代码的情况下，ObjectDataSource 使用中间层业务对象以声明方式对数据执行选择、插入、更新、删除、分页、排序、缓存和筛选操作。

ObjectDataSource 控件使用反射调用业务对象的方法，以对数据执行选择、更新、插入和删除操作。设置 ObjectDataSource 控件的 TypeName 属性来指定要用作源对象的类名称。

ObjectDataSource 控件包含 4 个重要属性：SelectMethod 属性、UpdateMethod 属性、InsertMethod 属性和 DeleteMethod 属性。综合利用这些属性，能够指定执行标准数据库操作所需的所有方法。

典型的数据访问层组件可按如下方式公开：

```
public class MyDataBllLayer {
public DataView GetRecords(){}
public int UpdateRecord(int recordID, String recordData){}
public int DeleteRecord(int recordID){}
public int InsertRecord(int recordID, String recordData){}
```

通常是在业务逻辑访问层定义对数据库里记录的操作，上面就定义了 GetRecords、UpdateRecord、DeleteRecord 和 InsertRecord 4 个方法来读取、更新、删除和插入数据库里的数据，这些方法基本上是根据 SQL 里的 SELECT、UPDATE、DELETE 和 INSERT 语句而定义的。

与方法相对应，ObjectDataSource 提供了 4 个属性来设置该控件引用的数据处理，可以按

照如下方式关联到该类型，代码如下：

```
<asp:ObjectDataSource TypeName="MyDataLayer"  runat="server"
   SelectMethod="GetRecords"
   UpdateMethod="UpdateRecord"
   DeleteMethod="DeleteRecord"
InsertMethod="InsertRecord"/>
```

这里的 SelectMethod 设置为 MyDataBllLayer 里的 GetRecords()方法，在使用时需要注意 ObjectDataSource 旨在以声明的方式简化数据的开发，所以这里设置 SelectMethod 的值为 GetRecords 而不是 GetRecords()。同样依此类推，UpdateMethod、DeleteMethod、InsertMethod 分别对应的是 UpdateRecord、DeleteRecord、InsertRecord 方法。

在定义 GetRecords()时，可以看到该方法返回的类型是 DataView，由于 ObjectDataSource 将来需要作为绑定控件的数据来源，所以它的返回类型必须是如下的返回类型之一：Ienumerable、DataTable、DataView、DataSet 或者 Object。

除此以外，ObjectDataSource 还有一个重要的属性 TypeName，ObjectDataSource 控件使用反射技术来从业务逻辑程序层的类对象调用相应的方法，所以 TypeName 的属性值就是用来标识该控件工作时使用的类名称。

四、拓展任务

本节完成三个拓展实践任务，两个课内拓展实践任务和一个课外拓展实践任务。

拓展任务卡 9

拓展任务号	5-9	任务名称	新闻列表页设计
计划用时	45 分钟	任务性质	课内
任务描述与目标			
企业网站前台除了首页外，还有其他一些页面，这些页面与首页的功能类似，主要是各类信息的展示。通过新闻列表页的设计与制作练习，熟练掌握数据源控件和数据绑定控件的使用			
主要操作步骤提示			
1．设计新闻列表页 Article.aspx，设计效果如图 5-14 所示；  			

图 5-14　新闻列表页设计效果

续表

拓展任务号	5-9	任务名称	新闻列表页设计
计划用时	45 分钟	任务性质	课内

2．编写后台事件代码，参考代码如下：

```
public string acnav { get; set; }
protected void Page_Load(object sender, EventArgs e)
{  string parentID = null;
   try
   {
     parentID = Request.QueryString["id"].ToString();
   }
   catch (Exception ex) { }
   if (BaseCommon.ValidQueryString(parentID))
   {
acnav = ArticleClassBll.ArticleClassNav(Convert.ToInt32(parentID), acnav);
     if(!IsPostBack)          {    this.ObjectDataSourceArticleClass
       .SelectParameters["parent_id"].DefaultValue = parentID;
        this.RepeaterArticleClass.DataBind();
 this.ObjectDataSourceArticle.SelectParameters["ac_id"].DefaultValue = parentID;
        AspNetPager1.RecordCount= ArticleBll.GetList(Convert.ToInt32(parentID)).Count;
     }
  }
  else
  {
     acnav = ArticleClassBll.ArticleClassNav(2, acnav);
     AspNetPager1.RecordCount = ArticleBll.GetList(2).Count;
    }
}  protected void RepeaterArticleClass_ItemDataBound

                (object sender, RepeaterItemEventArgs e)
{
    if (RepeaterArticleClass.Items.Count == 0)
    {
       this.RepeaterArticleClass.Visible = false;
    }
    else
    {
       this.RepeaterArticleClass.Visible = true;
    }
  }
  protected void ObjectDataSourceArticle_Selecting
               (object sender, ObjectDataSourceSelectingEventArgs e)
  {
    if (!e.ExecutingSelectCount)
    {
       e.Arguments.StartRowIndex = this.AspNetPager1.StartRecordIndex;
       e.Arguments.MaximumRows = this.AspNetPager1.PageSize;
    }
  }
```

拓展任务卡 10

拓展任务号	5-10	任务名称	新闻显示页设计
计划用时	15 分钟	任务性质	课内

任务描述与目标

通过新闻列表页，显示了企业的各类新闻，如果要查询某条新闻内容，则还需要开发新闻显示页，在新闻列表页中单击某条新闻的标题链接，就可转向新闻显示页，进行新闻的显示

主要操作步骤提示

1．设计新闻显示页 ArticleShow.aspx，设计效果如图 5-15 所示。

图 5-15　新闻显示页设计效果

2．编写后台事件代码，参考代码如下：

```
public string acnav { get; set; }
protected void Page_Load(object sender, EventArgs e)
{  if (!IsPostBack)
  {     Master.SetSiteProperty();
        string artID = null;
         try
            {
               artID = Request.QueryString["id"].ToString();
            }
            catch(Exception ex)
            {
            }
            Article myArticle = new Article();
            if (BaseCommon.ValidQueryString(artID))
            {
               ArticleBll.Update(Convert.ToInt32(artID));
               myArticle = ArticleBll.GetItem(Convert.ToInt32(artID));

               acnav = ArticleClassBll.ArticleClassNav(myArticle.ac_id, acnav);
               Page.Title = myArticle.art_title;
            HtmlMeta keywords= (HtmlMeta)Master.FindControl("keywords");
         HtmlMeta Description = (HtmlMeta)Master.FindControl("Description");
         keywords.Attributes["content"] = keywords.Attributes["content"] + "," + myArticle.
art_title;
       Description.Attributes["content"] = Description.Attributes["content"] + "," + myArticle.
art_description;
               this.lblTitle.Text = myArticle.art_title;
               this.lblDate.Text = myArticle.art_date.ToString();
               this.lblAuthor.Text = myArticle.art_author;
               this.lblFrom.Text = myArticle.art_from;
               this.lblClick.Text = myArticle.art_click.ToString();
               this.lblContent.Text = myArticle.art_content;
            }
            else
            {
               document.location.href='ArticleList.aspx'</script>");
                Response.Write("错误的链接!");
            }
        }
 }
```

拓展任务卡 11

拓展任务号	5-11	任务名称	企业网站前台其他页面设计
计划用时	240 分钟	任务性质	课外
任务描述与目标			
企业网站前台除了首页外，还有其他一些页面，这些页面与首页的功能类似，主要是各类信息的展示。通过新闻列表页的设计与制作练习，熟练掌握数据源控件和数据绑定控件的使用			
主要操作步骤提示			

1．设计产品列表页 ProductList.aspx，设计效果如图 5-16 所示。

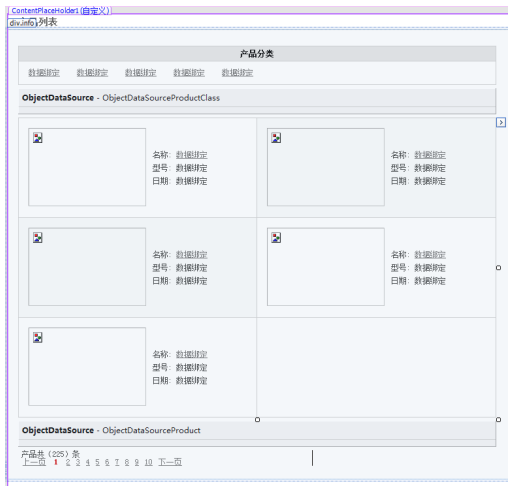

图 5-16　产品列表页设计效果

2．编写后台事件代码，代码请参见项目源代码。

3．设计产品显示页 ProductShow.aspx，设计效果如图 5-17 所示。

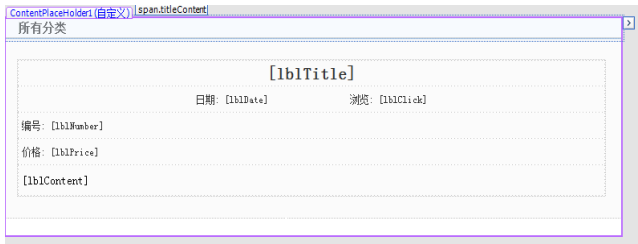

图 5-17　产品显示页设计效果

4．编写后台事件代码，代码请参见项目源代码。

5．设计公司概况页 About.aspx，设计效果如图 5-18 所示。

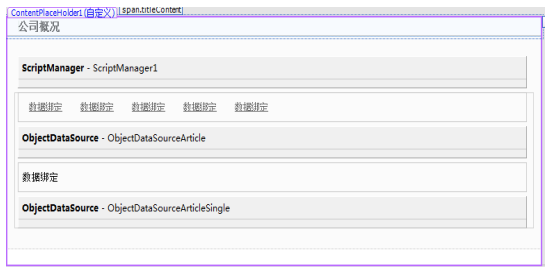

图 5-18　公司概况页设计效果

续表

拓展任务号	5-11	任务名称	企业网站前台其他页面设计
计划用时	240 分钟	任务性质	课外

6．编写后台事件代码，代码请参见项目源代码。

7．编写意见反馈页 Feedback.aspx，设计效果如图 5-19 所示。

图 5-19　意见反馈页设计效果

8．编写后台事件代码，代码请参见项目源代码

5.5.3　GridView 数据控件

一、实战演练

（1）利用后台管理模板页，新建一个新闻文章管理页 ArticleMng.aspx，在 ContentPlaceHolder 中添加一个 GridView 控件数据库显示控件和一个 ObjectDataSource 数据源控件，设计效果如图 5-20 所示。

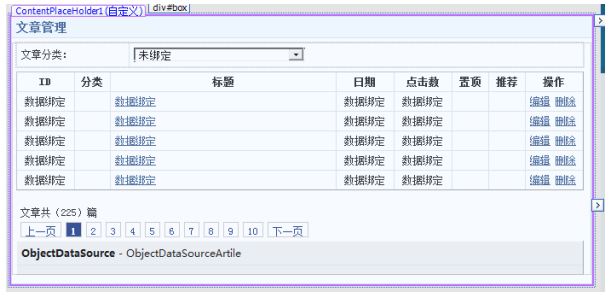

图 5-20　新闻文章管理页

（2）完成后的 GridView 控件源代码如代码清单 5-16 所示。

代码清单 5-16　GridView 控件源代码

```
    <asp:GridView  ID="GdVArtile"  runat="server"  DataSourceID="ObjectDataSource
Artile"
     AutoGenerateColumns="False"DataKeyNames="art_id"
onrowdeleted="GridViewArtile_RowDeleted"
     onrowdatabound= "GridViewArtile_RowDataBound">
      <Columns>
        <asp:BoundField    DataField="art_id"  HeaderText="ID"  InsertVisible="
False" ReadOnly="True"
```

```
        SortExpression="art_id" >
         <ItemStyle HorizontalAlign="Center" />
        </asp:BoundField>
        <asp:TemplateField HeaderText="分类">
              <ItemTemplate>
                 <asp:Label ID="lblAcName" runat="server" Text=""></asp:Label>
              </ItemTemplate>
              <ItemStyle HorizontalAlign="Center" />
       </asp:TemplateField>
       <asp:HyperLinkField  DataNavigateUrlFields="art_id" DataNavigateUrlFormat
String= "~/ArticleShow. aspx?ID={0}"DataTextField="art_title"   HeaderText="标
题"
       SortExpression="art_title" Target="_blank" >
     <ItemStyle Width="40%" />
     </asp:HyperLinkField>
     <asp:BoundField  DataField="art_date" HeaderText="日期"
           SortExpression="art_date" DataFormatString="{0:d}" >
       <ItemStyle HorizontalAlign="Center" />
     </asp:BoundField>
     <asp:BoundField DataField="art_click" HeaderText="点击数" SortExpression="
art_click" >
         <ItemStyle HorizontalAlign="Center" />
     </asp:BoundField>
     <asp:TemplateField HeaderText="置顶">
        <ItemTemplate>
        <asp:Image  ID="imgTop"  runat="server"  ImageUrl="~/Admin/Images/tick.
png"Visible="False" />
        </ItemTemplate>
       <ItemStyle HorizontalAlign="Center" />
    </asp:TemplateField>
    <asp:TemplateField HeaderText="推荐" >
    <ItemTemplate>
      <asp:Image  ID="imgCommend"  runat="server"ImageUrl="~/Admin/Images/tick.
png"Visible="False" />
      </ItemTemplate>
     <ItemStyle HorizontalAlign="Center" />
    </asp:TemplateField>
      <asp:TemplateField HeaderText="操作">
       <ItemTemplate>
      <asp:LinkButton ID="lbtEdit" runat="server" Text="编辑" PostBackUrl=
    '<%# "ArticleEdit.aspx?id=" + Eval("art_id")%>' >
    </asp:LinkButton>
      <asp:LinkButton ID="lbtDelete" runat="server"  Text="删除" OnClientClick="
return confirm
    ('确认要删除这篇文章吗?');" CommandName="Delete">
    </asp:LinkButton>
       </ItemTemplate>
     <ItemStyle Width="10%" />
    </asp:TemplateField>
    </asp:GridView>
```

（3）编写文章管理页 ArticleMng.aspx 的后台事件处理代码，如代码清单 5-17 所示。

代码清单 5-17　文章管理页后台事件处理代码

```
    protected void Page_Load(object sender, EventArgs e)
    {
    if (!IsPostBack)
    {
       LoadTree();//加载文章类别树
```

```
        //获取第三方分页控件 AspNetPager 的总记录数属性
        AspNetPager1.RecordCount=ArticleBll.GetList(Convert.ToInt32( this.Drop
        DownListTree.SelectedValue)).Count;
    }
    }
    protected void ObjectDataSourceArtile_Selecting(object sender, ObjectData
SourceSelectingEventArgs e)
    {
    if (!e.ExecutingSelectCount)
    {
        e.Arguments.StartRowIndex = this.AspNetPager1.StartRecordIndex;
        e.Arguments.MaximumRows = this.AspNetPager1.PageSize;
     }
      }
    protected void LoadTree()
    {
    this.DropDownListTree.Items.Clear();
    DataSet ds = ArticleClassBll.GetTree();
    string strName = null;
    for (int i = 0; i < ds.Tables["ArticleClass"].Rows.Count; i++)
      {
       strName = null;
       for(int j = 0; j <Convert.ToInt32(ds.Tables["ArticleClass"].Rows[i]["
depth"]);j++)
        {
             strName = strName + (char)0xa0 + (char)0xa0 + (char)0xa0;
         }
    strName = strName + ">" + ds.Tables["ArticleClass"].Rows[i]["ac_name"].
ToString();
        ListItem li = new ListItem(strName,
     ds.Tables["ArticleClass"].Rows[i]["ac_id"].ToString());
         this.DropDownListTree.Items.Insert(i, li);
    }
    }
    protected void DropDownListTree_SelectedIndexChanged(object sender, Event
Args e)
    {//依据分类列表的选项，确定该分类下的文章数，并写入到分页控件的 RecordCout 属性中
          AspNetPager1.RecordCount = ArticleBll.GetList
    (Convert.ToInt32(this.DropDownListTree.SelectedValue)).Count;
     }
    protected void GridViewArtile_RowDeleted(object sender, GridViewDeleted
EventArgs e)
     {
    //删除某个分类下的文章后，刷新分页控件的总记录属性
          AspNetPager1.RecordCount = ArticleBll.GetList
    (Convert.ToInt32(this.DropDownListTree.SelectedValue)).Count;
      }
    protected void GridViewArtile_RowDataBound(object sender, GridViewRow
EventArgs e)
    {
       if (e.Row.RowType == DataControlRowType.DataRow)
       {//根据文章是否是最新、推荐类型，如果是，则在表中显示对应的图标
          Image imgt = (Image)e.Row.FindControl("imgTop");
          Image imgc = (Image)e.Row.FindControl("imgCommend");
          Label lblAcName = (Label)e.Row.FindControl("lblAcName");
          Article myArticle = ((Article)e.Row.DataItem);
          if (myArticle.istop)
          {
             imgt.Visible = true;
```

```
            }
            if (myArticle.iscommend)
            {
                imgc.Visible = true;
            }
 //确定选中文章的文章类别
            int artID = (Int32)GridViewArtile.DataKeys[e.Row.RowIndex].Value;
            int acID = ArticleBll.GetItem(artID).ac_id;
            lblAcName.Text = ArticleClassBll.GetItem(acID).ac_name;
        }
 }
```

（4）运行测试。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-25 所示。

表 5-25　演练完成情况评价表

任务号	5-12	任务名称	开发新闻文章管理页
任务子项	完成情况	主要问题	
设计新闻文章管理页			
编写后台处理代码			

三、知识点

GridView 控件用于显示数据表中的数据。通过 GridView 控件，用户可以显示、编辑、删除和排序多种不同的数据源（包括数据库、XML 文件和公开数据的业务对象等）中的数据。

GridView 控件的常用属性和方法如下：

✧ AllowPaging 属性：获取或设置一个 Bool 型的值，用于表示是否启用该控件的分页功能，true 表示启用，false 表示不启用，默认值为 false。注意：启用分页功能时，从数据源控件返回的数据必须是 DataSet 类型；

✧ AllowSorting 属性：获取或设置一个值，该值指示是否启用排序功能；

✧ Columns 属性：获取表示 GridView 控件中列字段的 DataControlField 对象的集合；

✧ DataSource 属性：获取或设置对象，数据绑定控件从该对象中检索其数据项列表；

✧ DataSourceID 属性：获取或设置控件的 ID，数据绑定控件从该控件中检索其数据项列表；

✧ EditIndex 属性：获取或设置要编辑的行的索引；

✧ GridLines 属性：获取或设置 GridView 控件的网格线样式；

✧ PageIndex 属性：获取或设置当前显示页的索引；

✧ Rows 属性：获取表示 GridView 控件中数据行的 GridViewRow 对象的集合；

✧ SelectedIndex 属性：获取或设置 GridView 控件中的选中行的索引；

✧ SelectedRow 属性：获取对 GridViewRow 对象的引用，该对象表示控件中的选中行；

✧ SelectedValue 属性：获取 GridView 控件中选中行的数据键值；

✧ DataBind()方法：将数据源绑定到 GridView 控件；

✧ DeleteRow()方法：从数据源中删除位于指定索引位置的记录；

✧ FindControl()方法：在当前的命名容器中搜索指定的服务器控件；

✧ Focus()方法：为控件设置输入焦点；

✧ Sort()方法：根据指定的排序表达式和方向对 GridView 控件进行排序；

✧ UpdateRow()方法：使用行的字段值更新位于指定行索引位置的记录。

DataBind 方法用来执行数据库绑定操作，其常用语法如下：

```
string strCon = "Data Source=(local);Database=Northwind;User id=sa;Pwd=";
SqlConnection sqlconn = new SqlConnection(strCon);
string sqlstr = "select * from Region";
sqlconn.Open();
SqlDataAdapter myda = new SqlDataAdapter(sqlstr, sqlconn);
DataSet myds = new DataSet();
myda.Fill(myds);
GridView1.DataSource = myds;
GridView1.DataBind();
sqlconn.Close();
```

GridView 控件的常用事件如下:

✧ PageIndexChanged 事件：在单击某一页导航按钮时，GridView 控件处理分页操作之后发生；

✧ PageIndexChanging 事件：在单击某一页导航按钮时，GridView 控件处理分页操作之前发生；

✧ RowCancelingEdit 事件：单击编辑模式中某一行的“取消”按钮以后，在该行退出编辑模式之前发生；

✧ RowCommand 事件：当单击 GridView 控件中的按钮时发生；

✧ RowCreated 事件：在 GridView 控件中创建行时发生；

✧ RowDataBound 事件：在 GridView 控件中将数据行绑定到数据时发生；

✧ RowDeleted 事件：单击某一行的“删除”按钮时，GridView 控件删除该行之后发生；

✧ RowDeleting 事件：单击某一行的“删除”按钮时，GridView 控件删除该行之前发生；

✧ RowEditing 事件：在单击某一行的“编辑”按钮以后，GridView 控件进入编辑模式之前发生；

✧ RowUpdated 事件：在单击某一行的“更新”按钮，并且 GridView 控件对该行进行更新之后发生；

✧ RowUpdating 事件：在单击某一行的“更新”按钮以后，GridView 控件对该行进行更新之前发生；

✧ SelectedIndexChanged 事件：在单击某一行的“选择”按钮，GridView 控件对相应的选择操作进行处理之后发生；

✧ SelectedIndexChanging 事件：在单击某一行的“选择”按钮以后，GridView 控件对相应的选择操作进行处理之前发生；

✧ Sorted 事件：单击用于列排序的超级链接时，对相应的排序操作进行处理之后发生；

✧ Sorting 事件：单击用于列排序的超级链接时，对相应的排序操作进行处理之前发生。

如果要在 GridView 控件的某个事件下实现功能，可以在属性对话框中单击  按钮，找到相应事件，然后双击进入其后台页中编写代码。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 12

拓展任务号	5-12	任务名称	文章管理其他页面设计
计划用时	200 分钟	任务性质	课外
任务描述与目标			
企业网站后台中涉及文章管理功能的页面还有文章添加页，文章分类管理、文章分类添加等页面。除了文章管理之外，还包括产品管理、产品分类管理、产品添加和产品类别添加等			
主要操作步骤提示			
1．在项目中创建文章添加页，编写事件后台代码； 2．创建文章分类添加页，编写后台事件代码； 3．创建文章分类管理页，编写后台事件代码； 4．添加产品管理页面； 5．添加用户意见反馈信息管理			

5.6 任务 5 ASP.NET 页面安全设置

任务描述

在企业网站中，普通用户只能访问网站的前台，而网站的后台，只有管理员登录后才能进入到后台管理界面。通过本任务将学习如何保证网页的安全。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）使用 ASP.NET 的身份验证和授权机制实现网站的安全性；

一、实战演练

（1）在项目中添加 Admin 目录，此目录用来存放只有管理员才能访问的文件。

（2）打开项目根目录下的 web.config 文件，修改对应的配置节，代码如下：

```
    <authentication mode="Forms">
      <forms name="mycookiename" loginUrl="login.aspx" protection="All" timeout="
30">
      </forms>
    </authentication>
    <authorization>
      <allow users="*"/>
    </authorization>
```

（3）在 Admin 目录下添加配置文件 web.config，打开该文件，添加如下节点：

```
    <authorization>
       <deny  users="?"/>
     </authorization>
```

（4）在 Admin 目录下新建一个登录文件 login.asp，用于登录后台管理系统。后台管理系统登录页面设计如图 5-21 所示。

图 5-21　后台管理系统登录页面设计

（5）双击“登录”按钮，添加登录功能代码，如代码清单 5-18 所示。

代码清单 5-18　登录功能代码

```
    protected void btnLogin_Click(object sender, EventArgs e)
    {
        //调用管理员业务类 Admin 的 GetItem 方法，进行身份验证
       Admin myAdmin = AdminBll.GetItem(this.tbxUsername.Text.Trim(), FormsAuthentica
tion. HashPasswordForStoringInConfigFile(this.tbxPassword.Text.Trim(), "md5"));
        if (string.IsNullOrEmpty(myAdmin.username))
        { //验证失败
            this.lblError.Visible = true;
        }
        else
        {//通过验证，获取管理员的最后登录时间和登录 IP 地址信息，而后重定向到后台首页
          this.lblError.Visible = false;
          AdminBll.Update(myAdmin.admin_id,
Request.ServerVariables["REMOTE_ADDR"], DateTime. Now);
          Session["admin_user"] = this.tbxUsername.Text.Trim();
          Response.Redirect("Default.aspx");
        }
    }
```

（6）打开后台母版页 Admin.master 的后台文件，修改管理员注销处理代码。

```
    FormAuthentication.SignOut();
    Response.Redirect("Default.aspx");
```

（7）重新生成解决方案，测试网站的安全性。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-26 所示。

表 5-26　演练完成情况评价表

任务号	5-13	任务名称	ASP.NET 页面安全设置
任务子项	完成情况	主要问题	未完成原因
修改根目录下的 Web.config			
在 Admin 目录下添加 Web.config，并配置			
修改后台管理员登录和注销代码			
测试站点的安全性			

三、知识点

Web 站点中创建的页面用于供用户浏览访问。这些页面可以分为两种类型：一种是允许所

有用户访问，即无论用户是注册用户还是普通用户均可以访问页面；另一种是只允许部分用户访问。这部分用户必须向 Web 站点提交用户凭证才能够访问受限资源。对于后者，主要涉及两个重要概念，即身份验证和授权。身份验证是指确定请求用户身份的行为。通常，用户需向服务器提交凭证以便进行身份验证。一旦用户凭证通过验证，就必须确定此用户是否可以访问特定资源，这个过程被称为授权。

1. ASP.NET 的认证方式

ASP.NET 提供了三种认证方式：Windows 认证、Forms 身份验证和客户证书身份验证。

ASP.NET 身份验证的设置是比较简单的，通过对 web.config 文件的 authentication 配置节的 mode 属性可以完成设置，设置代码如下：

```
<system.web>
   <authentication mode="[Form|Passport|None]">
   <forms>...</forms>
   <passport/>
   </authentication>
</system.web>
```

由于 Windows 认证主要是基于局域网内的 Windows 用户，而客户证书身份认证是需要付费的，因而在互联网中 Forms 身份验证就成为首选验证方式。

Forms 身份验证是通过用户所创建的登录窗体来验证用户的用户名和密码。如果应用程序对请求通过了验证，系统会颁发一个票证，该票证包含用于重建后续请求的标识的密钥。Forms 身份验证流程如图 5-22 所示。

在 web.config 文件中，所有与 Forms 验证有关的设置均被放置在 forms 配置节中，因此，了解 forms 配置节的内容有助于更好地应用 Forms 验证。forms 配置节声明代码如下：

```
<forms name="name"  loginUrl="URL" defaultUrl="URL" >
       <credentials>...</credentials>
</forms>
```

forms 配置节的常用属性如下：

✧ Cookieless 属性：可选的属性，定义是否使用 Cookie 及 Cookie 的行为；

✧ DefaultUrl 属性：可选的属性，定义在身份验证之后用于重定向的默认 URL；

✧ LoginUrl 属性：可选的属性，指定如果找不到任何有效的身份验证 Cookie，将请求重定向到的用于登录的 URL。

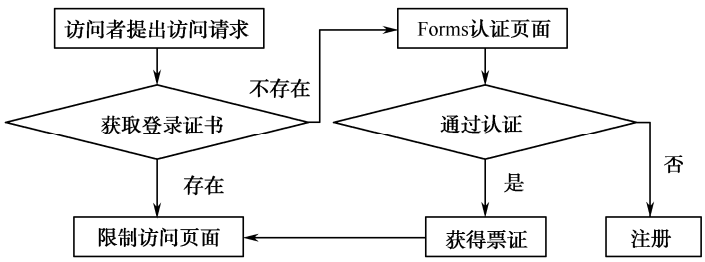

图 5-22　Forms 身份验证流程

2. authorization 配置节

该配置节为 Web 应用程序配置授权，以控制对 URL 资源的客户端访问。

authorization 配置节的声明语法如下：

```
<authorization>
  <allow .../>
  <deny .../>
</authorization>
```

authorization 配置节的子元素如下：

✧ allow 子标记：向授权规则映射添加一个规则，该规则允许对资源进行访问；

✧ deny 子标记：向授权规则映射添加一条拒绝对资源的访问的授权规则。

运行时，授权模块从最本地的配置文件开始，循环访问 allow 和 deny 元素，直到它找到适合特定用户账户的第一个访问规则。然后，该授权模块根据找到的第一个访问规则是 allow 还是 deny 规则来允许或拒绝对 URL 资源的访问。默认的授权规则为 <allow users="*"/>。

authorization 配置节的 allow 和 deny 子元素都有三个属性，即 users、roles、verbs。users 属性值是一个逗号分隔的用户名列表，也可以用“?”、“*”来代替，“?”指示授予匿名用户对资源的访问权，“*”指示授予所有用户对资源的访问权。roles 的属性值是一个逗号分隔的角色列表。verbs 的属性值是一个逗号分隔的 HTTP 传输方法列表，这些 HTTP 传输方法被授予对资源的访问权限。

3. FormsAuthentication 类

FormsAuthentication 类主要用于为 Web 应用程序管理 Forms 身份验证服务。该类提供了相应的方法和属性，可以在需对用户进行身份验证的应用程序中使用。

FormsAuthentication 类的常用属性和方法如下：

✧ CookieDomain 属性：获取 Forms 身份验证 Cookie 域的值；

✧ CookieMode 属性：获取一个值，该值指示是否已将应用程序配置为无 Cookie 的 Forms 身份验证；

✧ DefaultUrl 属性：获取在没有指定重定向 URL 时 FormsAuthentication 类将重定向到的 URL；

✧ FormsCookieName 属性：获取用于存储 Forms 身份验证票证的 Cookie 名称；

✧ FormsCookiePath 属性：获取 Forms 身份验证 Cookie 的路径；

✧ LoginUrl 属性：获取 FormsAuthentication 类将重定向到的登录页的 URL；

✧ Authenticate()方法：对照存储在应用程序配置文件中的凭据来验证用户名和密码；

✧ GetAuthCookie()方法：为给定的用户名创建身份验证 Cookie；

✧ GetRedirectUrl()方法：返回导致重定向到登录页的原始请求的重定向 URL；

✧ RedirectToLoginPage()方法：将浏览器重定向到登录 URL；

✧ SetAuthCookie()方法：为提供的用户名创建一个身份验证票证，并将其添加到响应的 Cookie 集合或 URL；

✧ SignOut()方法：从浏览器删除 Forms 身份验证票证。

4. 成员资格管理

ASP.NET 成员资格提供了一种验证和存储用户凭据的内置方法。ASP.NET 成员资格可管理网站中的用户身份验证。将 ASP.NET 成员资格与 ASP.NET Forms 身份验证或 ASP.NET 登录控件一起使用可以创建一个完整的用户身份验证系统。

四、任务拓展

本小节主要完成一个课内拓展任务。

拓展任务卡 13

拓展任务号	5-13	任务名称	利用 ASP.NET 成员资格管理网站中的用户身份验证
计划用时	20 分钟	任务性质	课内

任务描述与目标

.NET 提供了一种验证和存储用户凭证的内置方法——ASP.NET 成员资格，它可以帮助管理网站中的用户身份验证。在本次任务中主要完成为成员资格创建账户和利用成员资格重写管理登录与注销功能。通过练习掌握成员资格在网站身份验证中的应用

主要操作步骤提示

1．在 VS 2012 平台下，执行“菜单”→“网站”→“ASP.NET 配置”命令，将打开 ASP.NET 网站管理工具，如图 5-23 所示。

图 5-23　ASP.NET 网站管理工具

2．在窗口中选择“安全”选项卡，将看见“创建用户”链接，如图 5-24 所示。

3．单击“创建用户”项目，将打开创建新用户的类似向导界面，如图 5-25 所示。

图 5-24　“安全”选项卡

图 5-25　创建用户向导界面

4．填写表单项后，单击“创建用户”按钮，如果成功，则创建了一个用户账户。

5．利用 FormsAuthentication 类与 Membership 类重写网站后台管理员登录和注销功能，参考代码如下：

```
//用户登录
  If(Membership.ValidateUser(txtUsername.Text,txtPassword.Text))
    {
     FormsAuthentication.RedirectFromLoginPage(txtUserName.Text,false);
    }
//登录注销
  FormsAuthentication.SignOut();
  Response.Redirect("Default.aspx");
```

5.7 任务 6 建立与其他应用程序间的通信

任务描述

在企业中，经常都要把用不同语言写成的、在不同平台上运行的各种程序集成起来，而这种集成将花费很大的开发力量。通过 Web 服务可以实现跨平台间的互操作。本节的任务是对通信录项目进行改造，利用 Web 服务为企业网站项目提供员工信息服务。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）创建 Web 服务；

（2）在项目中使用 Web 服务；

（3）了解 WCF 技术。

5.7.1 创建通信录 Web Service

下面通过一个示例具体介绍如何创建 Web 服务。该 Web 服务具备查询功能，实现按指定员工号，查询员工联系电话的功能。

一、实战演练

（1）在 Visual Studio 2012 开发环境中，打开项目三中的 Address 网站，右击项目名，选择“添加”→“添加新建项目”。在如图 5-26 所示的“添加新项”对话框中选择“Web 服务”模板，并命名为“AddressWebService”。

（2）单击“确定”按钮，在新建的项目中自动生成 AddressWebService.asmx 文件，并自动生成初始代码，代码格式如代码清单 5-19 所示。

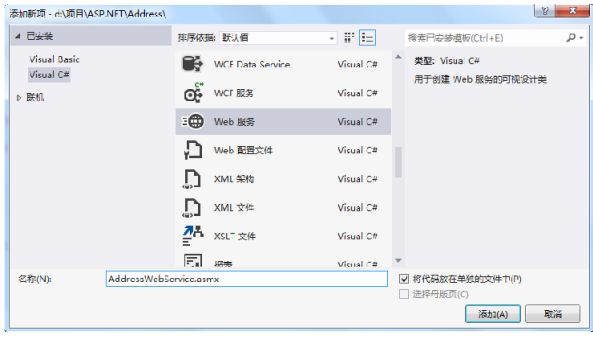

图 5-26 “添加新项”对话框

代码清单 5-19 Service1.asmx.cs 文件的初始代码

```
namespace AddressWebService
{
    // <summary>
    // Service1 的摘要说明
```

```
        // </summary>
        [WebService(Namespace = "http://tempuri.org/")]
        [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
        [System.ComponentModel.ToolboxItem(false)]
        // 若要允许使用 ASP.NET Ajax 从脚本中调用此 Web 服务，请取消对下行的注释
        // [System.Web.Script.Services.ScriptService]
        public class Service1 : System.Web.Services.WebService
        {
            [WebMethod]
            public string HelloWorld()
            {           return "Hello World";       }
       }}
```

（3）在代码中添加自定义的方法 GetContactinfoByUserName ( )。这个方法获取指定用户的通信录中的联系人，如代码清单 5-20 所示。

代码清单 5-20　自定义的方法 GetContactinfoByUserName( )代码

```
    [WebMethod]
        public DataSet GetContactinfoByUserName(string username)
        {
            SqlConnection sqlconn = new SqlConnection(@"server=.\SqlExpress;
database= addressbook; user id=*;Password=*");
            sqlconn.Open();
            SqlCommand cmd = new SqlCommand("select * from contactinfo where
username=@un", sqlconn);
            cmd.Parameters.Add("@un", SqlDbType.VarChar).Value = username;
            SqlDataAdapter sda = new SqlDataAdapter();
            sda.SelectCommand = cmd;
            DataSet ds = new DataSet();
            sda.Fill(ds);
            cmd.Dispose();
            sqlconn.Dispose();
            return ds;
        }
```

（4）在“生成”菜单中，选择“生成解决方案”选项，生成 Web 服务。为了测试生成的 Web 服务，直接单击“调试”按钮，将显示 Web 服务帮助页面，如图 5-27 所示。

（5）在图中看到的 Web 服务包含了一个方法： GetContactinfoByUserName 方法。单击 GetContactinfoByUserName 方法的链接，将显示它的测试页面，如图 5-28 所示。

图 5-27　Web 服务帮助页面

图 5-28　GetContactinfoByUserName 方法的测试页面

（6）在测试页中输入要查询的会员名，单击“调用”按钮即可调用 Web 服务的相应方法，其结果在浏览器中以 XML 方式显示，如图 5-29 所示。

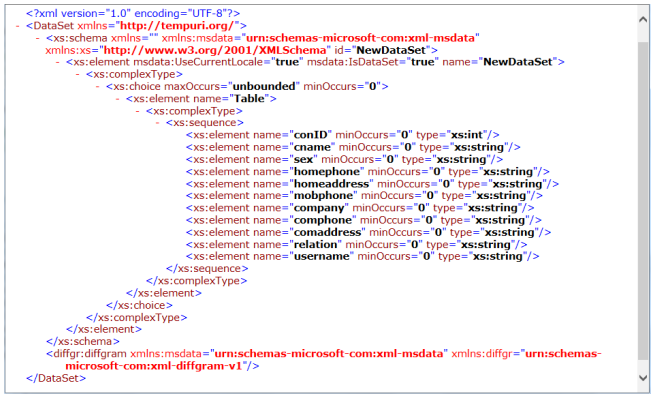

图 5-29　GetContactinfoByUserName 方法返回的结果页面

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-27 所示。

表 5-27　演练完成情况评价表

任务号	5-14	任务名称	建立与其他应用程序间的通信任务描述
任务子项	完成情况	主要问题	
创建 Web 服务			
添加 Web 服务方法			
测试 Web 服务			

三、知识点

Web Service 即 Web 服务。所谓服务就是系统提供一组接口，并通过接口使用系统提供的功能。在与 Windows 系统中通过 API 接口函数使用系统提供的服务一样，在 Web 站点之间，如果想要使用其他站点的资源，就需要其他站点提供服务，这个服务就是 Web 服务。

Web 服务是建立可互操作的分布式应用程序的新平台，它是一套标准，定义了应用程序如何在 Web 上实现互操作。在这个新的平台上，开发人员可以使用任何语言，以及任何操作系统平台上进行编程，只要保证遵循 Web 服务标准，就能够实现对服务进行查询和访问。Web 服务的服务端和客户端都要支持行业标准协议 HTTP、SOAP 和 XML。

Web 服务是让互联网上不同的工作区通过 HTTP 在互联网上传送 XML 的数据。因为传送的是 XML 的数据，所以一个组件只要安装在某一部计算机上即可。使用这种方式，可以节省下载、复制、安装和维护的时间，优点非常多。

每个 Web Service 都需要唯一的命名空间，它可使客户端应用程序区分可能使用相同的方法名称的 Web Service。在 Visual Studio .NET 中创建的 Web Service 的默认命名空间是“http://tempuri.org”。尽管命名空间类似于典型的 URL，但在 Web 浏览器中是不可能查看的，它只是一个唯一标识符。

Web 服务提供的属性如下：

✧ Description 属性：此属性的值包含描述性消息。此消息将在 XML Web Service 的说明文件（如服务说明和服务帮助页）生成后显示给 XML Web Service 的潜在用户；

✧ Name 属性：此属性的值包含 XML Web Service 的名称。在默认情况下，该值是实现 XML Web Service 的类的名称；

✧ Namespace 属性：此属性的值包含 XML Web Service 的默认命名空间。

四、任务拓展

本节完成 1 个课外拓展实践任务。

拓展任务卡 14

拓展任务号	5-14	任务名称	创建一个指定用户的分类查询联系人的自定义方法
计划用时	50 分钟	任务性质	课外
任务描述与目标			
创建 Web 服务方法，用于更新指定员工号的员工联系电话			
主要操作步骤提示			
1．在上一个服务的代码中添加自定义的方法 string GetContactinfoByrelation (string，contName,string username)。 2．测试生成的 Web 服务，直接单击按钮，将显示 Web 服务帮助页面。 3．单击 GetContactinfoByrelation 方法的链接将显示它的测试页面。 4．在测试页中输入要查询的指定用户的指定联系人，单击“调用”按钮即可调用 Web 服务的相应方法，在浏览器中以 XML 的方式显示联系人所有的电话号码			

5.7.2 在 Web 网站中调用通讯录 Web Service

在.NET 中调用 Web 服务其实和创建 Web 服务一样简单。以下通过实例来讲解和练习在 ASP.NET 中如何调用之前创建的 Web 服务。

一、实战演练

（1）打开企业网站。

（2）在项目上单击鼠标右键，在弹出的快捷菜单中选择“添加服务引用”选项，弹出如图 5-30 所示的“添加服务引用”对话框。在“URL”中输入前面所创建的 XML Web 服务的地址，则会列出 Web 服务的测试页和方法描述，同时验证项目是否能够使用 Web 服务。

（3）在对话框右边的“Web 引用名”中输入“addressService”，单击“添加引用”按钮。此时的解决方案面板的项目中多了一个 App_WebReferences 目录，其目录结构如图 5-31 所示。自动生成的代理类就放在这里。

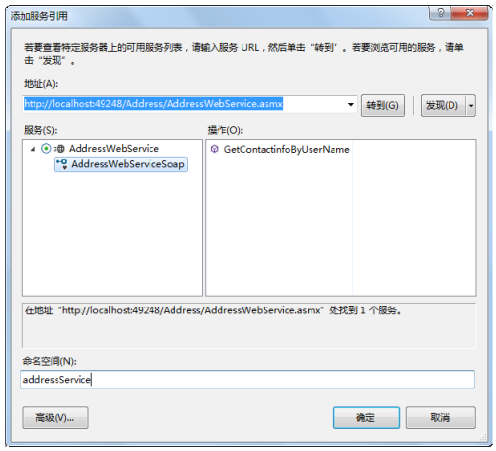

图 5-30 “添加服务引用”对话框

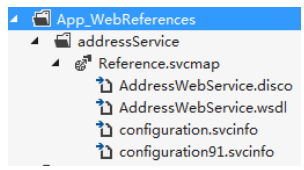

图 5-31 添加了 Web 引用后的目录结构

（4）在企业网站的后台管理中添加一个管理员登录功能后，就可以看到自己在网络通讯录中的所有联系人，代码如下。

```
    addressService.AddressWebServiceSoapClient addWeb  = new AddressWebService
SoapClient();
    DataSet ds = addWeb.GetContactinfoByUserName(Session["admin"].ToString());
    GridView1.DataSource = ds;
    GridView1.DataBind();
```

（5）到此为止，ASP.NET 调用 Web 服务的过程就完成了。运行测试，在页面中查看结果。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 5-28 所示。

表 5-28　演练完成情况评价表

任务号	5-15	任务名称	调用 Web 服务
完成情况		主要问题	未完成原因

三、任务拓展

本节完成 1 个课外拓展实践任务。

拓展任务卡 15

拓展任务号	5-15	任务名称	在 Internet 上查找可用的服务，为企业网站添加可用的功能
计划用时	50 分钟	任务性质	课外
任务描述与目标			
通过此任务，可以进一步理解 Web 服务的应用			
主要操作步骤提示			

1．在 Internet 上查找可用的 Web 服务，下面提供一些可用的 Web 服务地址：

http://webservice.webxml.com.cn/WebServices/WeatherWS.asmx，天气预报查询服务

http://fy.webxml.com.cn/webservices/EnglishChinese.asmx，中英文翻译服务

http://webservice.webxml.com.cn/WebServices/MobileCodeWS.asmx，国内手机号码归属地查询 Web 服务

http://webservice.webxml.com.cn/WebServices/StockInfoWS.asmx，股票行情数据 Web 服务（支持中国香港、深圳、上海基金、债券和股票；支持多股票同时查询）

通过 http://www.webxml.com.cn/zh_cn/web_services.aspx 这个网站可以查到更多的免费服务；

2．选择其中一个电话号码查询的 Web 服务添加在企业网站中，那么就可以在企业网站上看到 Internet 上其他网站提供的数据。在网站首页右下角添加一个号码查询功能，代码如下：

```
telephoneWeb.MobileCodeWS mws = new telephoneWeb.MobileCodeWS();
Label1.Text = mws.getMobileCodeInfo(txtTele.Text, "");
```

3．在页面中输入一个号码，可以查看到号码查询结果，如图 5-32 所示。

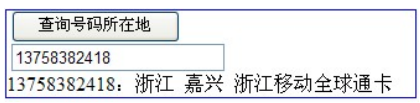

图 5-32　查询号码所在地

拓展任务跟踪卡

任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

项目号			项目名	
项目完成方式			□小组协作 □个人独立	
项目完成情况（40%）	界面设计（2%）	自我评价		
		小组评价		
		个人评价		
	代码编写（20%）	自我评价		
		小组评价		
		个人评价		
	功能实现（10%）	自我评价		
		小组评价		
		个人评价		
	说明文档（5%）	自我评价		
		小组评价		
		个人评价		
	数据库设计（3%）	自我评价		
		小组评价		
		个人评价		
项目展示（30%）	语言表达(10%)	自我评价		
		小组评价		
		个人评价		
	团队合作(10%)	自我评价		
		小组评价		
		个人评价		
	展示形象(10%)	自我评价		
		小组评价		
		个人评价		
拓展与创新（30%）	创新能力（10%）	自我评价		
		小组评价		
		个人评价		
	设计潜力（10%）	自我评价		
		小组评价		
		个人评价		
主要存在问题				

课外思考题

1．ascx 文件与 aspx 文件有何不同，如何创建和使用 ascx 文件？

2. 为什么要把项目数据库文件放入 App_Data 文件夹中，在程序中如何访问放在 App_Data 目录下的数据库？

3．什么是网站地图？如何快速创建网站地图？如何使用 TreeView 控件创建导航菜单？

4．如何限制对成员专用页面的访问，以实例说明。

5．Cookie 与 Session 有何联系与区别？

6．Global.asax 文件的作用是什么？Application 的 Lock 与 UnLock 方法有何用途？

7．Session_Start 事件与 Session_End 事件分别在何时触发?

8．创建一个名为 TestDataBinding.aspx 的示例 Web 窗体。在这个窗体中添加两个 ListBox 控件，将这两个控件绑定到两个不同的示例字符串数组。

数组 1：[“浙江省”,“福建省”]，数组 2：[“丽水”,“平湖”,“嘉兴”]

9．创建一个名为 TowColumns.master 的母版页，它包含标题，两个内容列和一个底部，这两个内容列中分别包含两个 Content 控件。创建一个使用母版页的测试页面。

10．创建一个名为 Authors.aspx 的页面，它在一个 DataList 中显示 Authors 表中的所有记录。列表应显示作者的姓名，作者的姓名是一个链接，用户通过该链接可以转到名为 AuthorBooks.aspx 的页面。AuthorId 应该通过一个名为 author 的查询字符串参数传递给 AuthorBooks.aspx。

11．创建一个名为 AuthorBooks.aspx 的页面，它使用一个 Repeater 来显示名为 author 的查询字符串参数所指定作者的所有图书。这个页面应该在标题中显示作者的名字，然后是每本图书的 ISBN、书名、封面、分类名、出版社及描述。

12．创建一个名为 ArtWorks.aspx 的页面，使用一个 GridView 控件来显示艺术品的标题、作者年份、定价、成本及艺术家的名字。这个 GridView 必须能够进行分页、排序和编辑。

13．在 12 题中，把一个 ButtonField 添加到 GridView 中，利用该按钮将一行艺术品信息从表中删除。

项目六

Ajax 聊天室

知识目标

通过对本项目的学习，应该掌握下面的知识：

- ✧ 掌握 ASP.NET Ajax 基础知识与结构；
- ✧ 掌握 ASP.NET Ajax 技术实现的常用控件 ScriptManager 的使用；
- ✧ 掌握 ASP.NET Ajax 技术实现的常用控件 Timer 的使用；
- ✧ 掌握 ASP.NET Ajax 技术实现的常用控件 UpdatePanel 的使用。

技能目标

能根据需要，在项目中使用 ASP.NET Ajax 服务端控件实现 Ajax 技术。

教学建议

本项目计划总学时为 12 学时。

- ✧ 情境介绍：1 学时；
- ✧ 任务 1：5 学时；
- ✧ 任务 2：6 学时。

教师在开展本项目教学之前，可以让学生就当前人们的信息沟通方式进行分组讨论，引出网络聊天室，通过在网络聊天室里进行聊天的亲身体验，探讨聊天室项目的需求。

6.1 情境介绍

随着互联网的迅速发展，网络聊天已经成为人与人之间沟通的主要桥梁之一，越来越多的人开始选择通过网络进行即时沟通。据权威报道，当前中国使用 QQ 进行信息沟通的同时在线人数已经过亿。除了 QQ 外，现在还有很多网站提供了在线聊天服务，如 163 网易聊天室等，这些聊天室多数同时支持视频、语言、文字聊天方式。本项目所要完成的 MiniAjax 聊天室就是一个支持文字聊天的简易 Web 聊天工具，注册为会员后就可以选择房间进入并进行聊天。

1. 聊天室的需求分析

在线聊天室的作用是在互联网上为用户提供一个聊天沟通的场所。为此，在线聊天室要具有以下功能：

- ✧ 管理员登录后可进行聊天室、会员的管理；
- ✧ 普通访问者可以注册成为会员，会员登录时可选择房间；
- ✧ 在聊天室里具备在线会员、聊天消息即时刷新；
- ✧ 会员可以进行聊天；
- ✧ 支持信息的异步刷新。

聊天室项目中涉及两种不同的角色，分别为管理员和注册会员，其中管理员主要是进行聊天室的管理工作；而为注册会员提供的主要是选择聊天室进行聊天业务。如图 6-1 和图 6-2 所示分别展示了聊天室管理员和注册会员的业务流程。

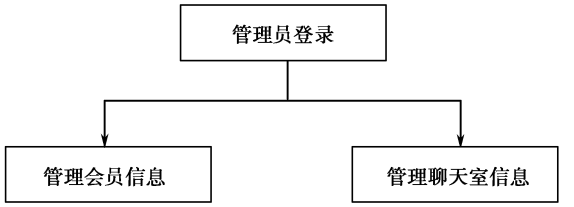

图 6-1 管理员的业务流程

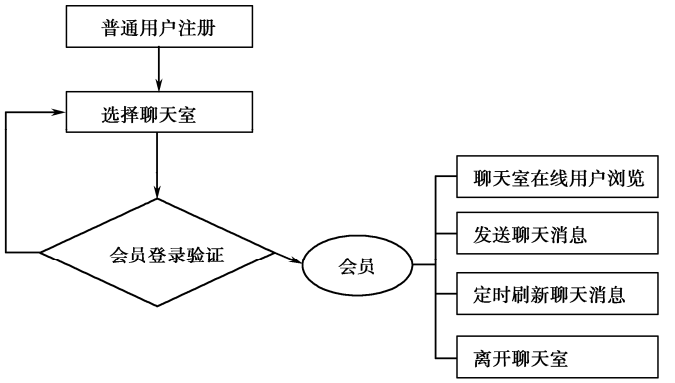

图 6-2 注册会员的业务流程

2. 聊天室功能模块

根据聊天室业务，主要功能分为两大模块，即前台和后台。其中前台模块功能包括选择聊天室、会员的注册和登录、在线聊天。在线聊天又分在线用户浏览、发送聊天消息、定时刷新

聊天消息和离开聊天室；后台的功能模块包括管理员的登录与注销、会员管理和聊天室管理。聊天室的功能结构如图 6-3 所示。

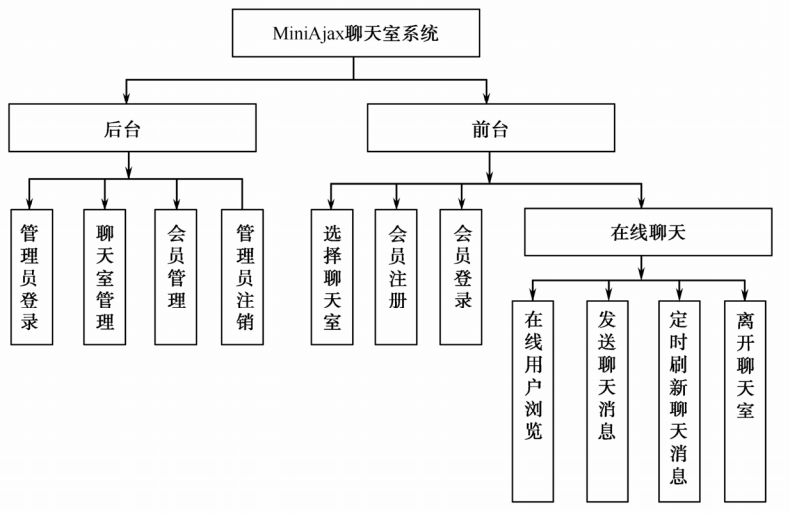

图 6-3　聊天室的功能结构

3. 网络聊天室的数据库设计

在 SQL Server 2005 中创建数据库 MChat_DB，在其中新建三个表，分别为 Table_ChatRoom、Table_User、Table_Message。其中，Table_ChatRoom 表用于保存聊天室的信息，Table_User 表用于保存用户的信息，Table_Message 表用于保存聊天的信息。下面创建所需的三张表。

（1）Table_ChatRoom 表，该表保存了聊天室的信息，它包含的字段及其说明如表 6-1 所示。

表 6-1　Table_ChatRoom 表

字 段 名	字 段 类 型	字 段 说 明	备　注
ChatRoom_ID	int	聊天室 ID	PK（自动增 1）
ChatRoom_Name	nvarchar(50)	聊天室名称	
ChatRoom_Status	bit	状态	1:开放 0:关闭
ChatRoom_MaxNum	tinyint	最大聊天人数	

（2）Table_User 表，该表保存了用户的信息，它包含的字段及其说明如表 6-2 所示。

表 6-2　Table_User 表

字 段 名	数 据 类 型	字 段 说 明	备　注
User_ID	int	用户 ID	PK（自动增 1）
User_Name	nvarchar(50)	用户名称	
User_PassWord	varchar(32)	用户密码	
User_RegTime	datetime	注册时间	默认值 getdate()

（3）Table_Message 表，该表保存了聊天的信息，它包含的字段及其说明如表 6-3 所示。

4. 存储过程

设计系统的数据库视图和存储过程，通过任务练习掌握 SQL Server 2005 中视图和存储过

程的开发。

表 6-3 Table_Message 表

字 段 名	数 据 类 型	字 段 说 明	备 注
Message_ID	int	聊天消息 ID	Pk（自动增 1）
Message_Body	nvarchar(500)	聊天消息内容	
ChatID	int	所在聊天室的 ID	FK 引用 Table_ChatRoom 表的 ID 列
Message_From	int	发送消息的用户 ID	FK 引用 Table_User 表的 User_ID 列
Message_To	int	接收消息的用户 ID	FK 引用 Table_User 表的 User_ID 列
Message_PubTime	datetime	发送时间	默认值 getdate()

（1）设计一个视图 View_AllMessage 实现如下功能：显示所有的消息，消息中应该包括发送消息的会员名称、接收消息的会员名称、发送消息的时间、消息的内容。

（2）开发如下存储过程：

Pr_AddChatRoom：添加一条聊天室记录

Pr_AddMessage：添加一条聊天消息

Pr_AddUser：添加一条用户记录

Pr_AllChatRoom：显示所有聊天室信息

Pr_AllChatRoomOpen：显示所有开放状态的聊天室

Pr_GetTop20Messages：显示某个聊天室内最近的 20 条聊天消息

Pr_GetUser：获取某个用户信息

Pr_UpdateChatRoom：更新聊天室信息

（3）运行测试。

5. *安全方案设计*

（1）在 VS 2012 中新建一个网站项目，命名为 MAjaxChat；

（2）在项目的 App_Data 目录下添加数据库文件 MChat_DB.mdf；

（3）在项目中新建名称分别为 Admin 和 Member 的两个目录；

（4）配置网站的 Web.config，确保 Admin 目录下除了名为 Login.aspx 的登录页外，其他页面只允许通过认证和授权的管理员访问；

（5）配置网站的 Web.config，确保 Member 目录下的页面只允许通过认证和授权的管理员和注册会员访问。

6.2 任务 1 ASP.NET Ajax 服务器控件

任务描述

主要是通过实现聊天室的各项管理业务，涉及了管理员登录、创建新的聊天室、聊天室的更新、关闭开放聊天室等功能的实现来学习 ASP.NET Ajax 服务器控件。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）聊天室业务逻辑类的设计；

（2）管理员登录设计；

（3）能够使用 ASP.NET Ajax 服务端控件实现页面信息的局部更新。

6.2.1 聊天室业务逻辑类的设计

在整个聊天室项目中，采用了两层结构，即表现层和业务逻辑层。表现层主要由各个页面和用户控件构成，而业务逻辑层主要是由一些类文件构成，这些类实现聊天室管理和聊天业务的功能。

一、实战演练

（1）在 VS 2012 中新建类 ChatRoomInfo，用于封装一个聊天室的信息，具体代码如下：

```
public class ChatRoomInfo
{
        public int ChatRoom_ID { get; set; }
        public string ChatRoom_Name { get; set; }
        public byte ChatRoom_MaxNum { get; set; }
        public byte ChatRoom_CurrentNum { get; set; }
}
```

（2）新建类 ChatRoomInfoDAL 用于封装聊天室的管理业务，具体代码中封装了如下几个业务方法：

✧ public bool CreateChatRoom(string room_name, byte max_num,bool status)方法：主要实现创建聊天室的功能；

✧ public bool UpdateChatRoom(int chatRoom_ID, string room_name, bool status, byte maxnum)方法：主要实现更新指定 ID 聊天室信息的功能；

✧ public bool isExistChatRoom(string room_name)方法：主要实现判定指定名称的聊天室是否存在的功能；

✧ public DataSet GetAllChatRoom()方法：主要实现获取所有聊天室信息的功能；

✧ public List<ChatRoomInfo> GetAllOpenRoom()方法：主要实现获取所有开放状态的聊天室信息的功能；

✧ public bool DeleteChatRoom(int chatRoom_ID)方法：主要实现删除指定 ID 聊天室信息的功能。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-4 所示。

表 6-4 演练完成情况评价表

任务号	6-1	任务名称	业务逻辑类的设计
任务子项	完成情况	主要问题	未完成原因
新建类 ChatRoomInfo			
新建类 ChatRoomInfoDAL			

三、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 1

拓展任务号	6-1	任务名称	通用类的设计
计划用时	120 分钟	任务性质	课外
任务描述与目标			
设计系统中业务逻辑层中的其他实体类和业务类，开发全局应用程序类，掌握业务类的开发方法			
主要操作步骤提示			
1．创建用于封装用户信息的实体类 UserInfo； 2．创建 UserInfoDAL 类，实现如下基本操作，即用户的登录验证、用户的注册、用户信息的更新； 3．创建 MessageInfoDAL 类，实现如下基本操作，即获取指定聊天室的最近 20 条消息、发送一条指定的消息			

6.2.2 管理员登录

只有以管理员的身份登录系统，才可以对聊天室中的相关信息进行各类管理操作。

一、实战演练

（1）打开项目下 Admin 目录中的 login.aspx 页面，设计管理员登录窗口，如图 6-4 所示。

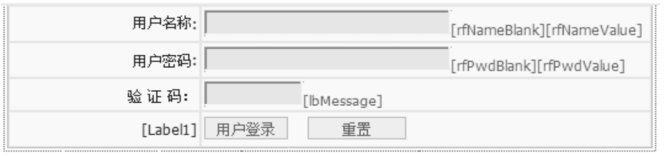

图 6-4 管理员登录窗口

（2）编写后台事件处理代码，实现管理员的登录功能，如代码清单 6-1 所示。

代码清单 6-1 管理员的登录功能代码

```
protected void btnLogin_Click(object sender,EventArgs e)
   {
     if(Session[ValidateCode.VALIDATECODEKEY] != null)
       {   //验证验证码是否正确
           if(tbCode.Text != Session[ValidateCode.VALIDATECODEKEY].ToString())
           {
               lbMessage.Text = "验证码输入错误，请重新输入";
               return;
           }
//判断管理员的密码和名称是否正确
if (FormsAuthentication.Authenticate(txtUserName.Text, txtPassWord.Text))
       {
       Session["admin"] = txtUserName.Text;
       FormsAuthentication.SetAuthCookie(txtUserName.Text, false);
       Response.Redirect("~/Admin/Default.aspx");
        }
else
     {
       Response.Redirect("Default.aspx");
}
    }
```

（3）运行测试。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价如表 6-5 所示。

表 6-5　演练完成情况评价表

任务号	6-2	任务名称	管理员登录
任务子项	完成情况	主要问题	未完成原因
管理员登录页面的设计			
后台事件代码的编写			
后台母版页的设计			
测试			

6.2.3　新建聊天室

注册会员进行登录聊天时，系统可供用户选择想要的聊天室并进入聊天，而聊天室的创建均是由管理员完成的。

一、实战演练

（1）在项目的 Admin 目录下新建 Web 窗口，命名为 AddChatRoom.aspx，该窗体用于填写要创建聊天室的相关信息。

（2）设计 AddChatRoom.aspx 窗口，设计效果如图 6-5 所示。

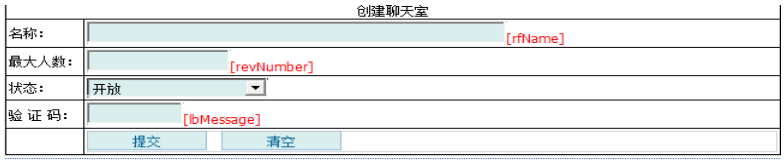

图 6-5　AddChatRoom.aspx 窗口的设计效果

（3）编写创建聊天室“提交”按钮的 Click 事件代码，如代码清单 6-2 所示。

代码清单 6-2　“提交”按钮的事件代码

```
    protected void btnSubmit_Click(object sender,EventArgs e)
       {
    if(Session[ValidateCode.VALIDATECODEKEY] != null)
       {    //验证验证码是否正确
           if(tbCode.Text != Session[ValidateCode.VALIDATECODEKEY].ToString())
           {
               lbMessage.Text = "验证码输入错误，请重新输入";
               return;
           }
         ChatRoomDAL chatRoomDAL = new ChatRoomDAL();
           //调用 ChatRoomDAL 类的 AddChat 方法实现添加新聊天室记录
        if
(chatRoomDAL.AddChat(tbName.Text,Int32.Parse(tbMaxNumber.Text),byte.Parse(
    ddlStatus.SelectedValue),tbRemark.Text) > 0)
        {
           //创建成功
           ScriptManager.RegisterClientScriptBlock
```

```
    (this, this.GetType(), String.Empty, "alert('创建成功！')", true);
    //转到聊天室管理页
          Response.Redirect("~/Amdin/ChatRoomMng.aspx");
    }
      else
      {
            //创建失败
        ScriptManager.RegisterClientScriptBlock(this,  this.GetType(),  String.
Empty, "alert('注册失败！')", true);
      }
    }
```

（4）测试运行。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-6 所示。

表 6-6　演练完成情况评价表

任务号	6-3	任务名称	创建聊天室
任务子项	完成情况	主要问题	未完成原因
创建聊天室页面的设计			
后台事件代码的编写			
测试			

6.2.4　ASP.NET Ajax 服务器控件

一、实战演练

（1）在项目的 Admin 目录下创建 Web 窗口，命名为 ChatRoomMng.aspx，该窗口是管理员进行管理聊天室的操作界面。

（2）在工具箱中选择 ScriptManager 控件拖入窗口。

（3）在工具箱中选择 UpdatePanel 控件拖入窗口。

（4）在 UpdatePanel 控件内拖入一个 GridView 控件，在 UpdatePanel 控件外拖入一个 Button 控件并设置控件的相关属性。

（5）完成后的 ChatRoomMng.aspx 的页面效果如图 6-6 所示。

（6）完成后的 ChatRoomMng.aspx 的关键 HTML 代码如代码清单 6-3 所示。

图 6-6　ChatRoomMng.aspx 的页面效果

代码清单 6-3　ChatRoomMng.aspx 的关键 HTML 代码

```
<asp:ScriptManager ID="ScriptManager1" runat="server"/>
  <asp:UpdatePanel ID="UpdatePanel1" runat="server">
      <ContentTemplate>
            <asp:GridView ID="ChatRoomGridView" ......>
            ....<!-- 此处省略 -->
            </asp:GridView>
```

```
                    </ContentTemplate>
                </asp:UpdatePanel>
```

（7）相关的后台事件代码如代码清单 6-4 所示。

代码清单 6-4　ChatRoomMng.aspx 的后台代码

```
    private void BindPageData()
        {   //获取聊天室数据
            ChatRoomDAL chatRoomDAL = new ChatRoomDAL();
            DataSet ds = chatRoomDAL.GetChats();
            //显示聊天室数据
            gvChat.DataSource = ds;
            gvChat.DataBind();
    }
        protected void btnAdd_Click(object sender,EventArgs e)
        {
            Response.Redirect("~/Admin/AddChatRoom.aspx");
        }
        protected void gvChat_RowCommand(object sender,GridViewCommandEventArgs e)
        {
            if(e.CommandName.ToLower() == "update")
            {   //重定向到修改页面
                Response.Redirect("~/UpdateChatRoom.aspx?ChatID="  +  e.Command
Argument.ToString());
                return;
            }
            if(e.CommandName.ToLower() == "del")
            {   //删除聊天室，并重新显示数据
                ChatRoomDAL chatRoomDAL = new ChatRoomDAL();

   if(chatRoomDAL.DeleteChat(Int32.Parse(e.CommandArgument.ToString())) > 0)
                {
                    BindPageData();
                }
                return;
            }
        }
        protected void gvChat_RowDataBound(object sender,GridViewRowEventArgs e)
        {       //添加删除时的确认对话框
            ImageButton imgDelete = (ImageButton)e.Row.FindControl("imgDelete");
            if(imgDelete != null)
            {   imgDelete.Attributes.Add("onclick","return
    confirm('您确认要删除当前行的聊天室吗?');");
            }
        }
        protected void gvChat_PageIndexChanging(object sender,GridViewPageEvent
Args e)
        {     //重新设置新页码
            gvChat.PageIndex = e.NewPageIndex;
            BindPageData();
        }
```

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-7 所示。

表 6-7 演练完成情况评价表

任务号	6-4	任务名称	管理聊天室
任务子项	完成情况	主要问题	未完成原因
聊天室管理页面的设计			
后台事件代码的编写			
测试			

三、知识点

1. Ajax 技术概述

Ajax 技术是基于 XML 的异步 JavaScript，简称 Ajax，是当前 Web（Web 2.0）创新中的一个王冠。

Ajax 并不是一门新的语言或技术，它实际上是几项技术按一定的方式组合在一起的，在共同的协作中发挥各自的作用，它包括：

- ✧ 使用 HTML 和 CSS 标准化呈现；
- ✧ 使用 DOM 实现动态显示和交互；
- ✧ 使用 XML 和 XSLT 进行数据交换与处理；
- ✧ 使用 XMLHttpRequest 进行异步数据读取；
- ✧ 使用 JavaScript 绑定和处理所有数据。

2. Ajax 的工作原理

Ajax 的工作原理相当于在用户和服务器之间加了一个中间层，使用户操作与服务器响应异步化。但并不是所有的用户请求都提交给服务器，如一些数据验证、数据处理等，都交给 Ajax 引擎来做，只有确定需要从服务器读取新数据时再由 Ajax 引擎代为向服务器提交请求。如图 6-7 所示给出了传统 Web 应用模式和基于 Ajax 的 Web 应用模式工作过程的区别。

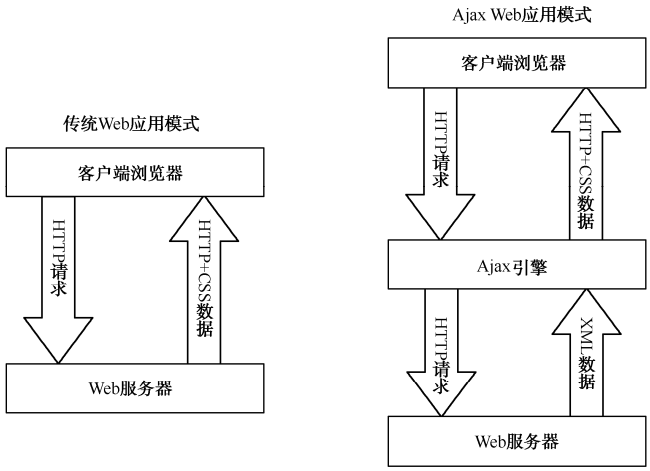

图 6-7 两种模式工作过程的区别

在旧的交互方式中，由用户触发一个 HTTP 请求到服务器，服务器对其进行处理后再返回一个新的 HTML 页到客户端。每当服务器处理客户端提交的请求时，客户只能空闲等待，并且哪怕只是一次很小的交互、只需从服务器端得到很简单的一个数据，都要返回一个完整的

HTML 页，而用户每次都要浪费时间和带宽去重新读取整个页面。使用 Ajax 后用户从感觉上认为几乎所有的操作都会很快响应，并且没有页面重载的等待。

3. Ajax 的优势

✧ 减轻服务器的负担，因为 Ajax 的根本理念是按需取数据，所以最大可能地减少了冗余请求和响应对服务器造成的负担；

✧ 无刷新更新页面，减少用户实际和心理等待时间；

✧ 更好的用户体验；

✧ 可以把以前的一些服务器担负的工作转移到客户端，利用客户端闲置的处理能力来处理，减轻服务器和带宽的负担，节约空间和带宽的租用成本；

✧ 可以调用外部数据；

✧ 基于标准化的并被广泛支持的技术，并且不需要插件或下载小程序；

✧ 使 Web 中的界面与应用分离（也可以说是数据与程序分离）；

✧ 对于用户和 ISP 来说是双赢的。

4. ASP.NET Ajax 控件

ASP.NET 内置了 5 个 Ajax 控件，分别是 ScriptManager、ScriptManagerProxy、UpdatePanel、UpdateProgress 和 Timer，其中 ScriptManager 、UpdatePanel 和 Timer 控件最常用。

1）ScriptManager 控件

ScriptManager 控件是 ASP.NET 中 Ajax 功能的中心，该控件可管理一个页面上的所有 ASP.NET Ajax 资源，其中包括将 Microsoft Ajax Library 脚本下载到浏览器和协调通过使用 UpdatePanel 控件启用的部分页面更新。ScriptManager 控件的常用语法如下：

```
<asp:ScriptManager ID="sm" runat="server" />
```

注意：一个页面在其层次结构中只能包含一个 ScriptManager 控件，若要在父页面已具有 ScriptManager 控件时为嵌套页面、用户控件或组件注册服务和脚本，则需使用 ScriptManagerProxy 控件。

2）UpdatePanel 控件

通过使用 UpdatePanel 控件，可以使网页参与到部分页更新中，而无须编写任何客户端脚本。UpdatePanel 控件在网页中需要使用 ScriptManager 控件。默认情况下，将启用部分页更新，因为 ScriptManager 控件的 EnablePartialRendering 属性的默认值为 true。

默认情况下，UpdatePanel 控件内的任何回发控件都将导致异步回发并刷新面板的内容。但是，也可以配置页面上的其他控件来刷新 UpdatePanel 控件，可以通过为 UpdatePanel 控件定义触发器来做到这一点。触发器是一类绑定，用于指定使面板更新的回发控件和事件。当引发触发器控件的指定事件（如按钮的 Click 事件）时，将刷新更新面板。

使用 UpdatePanel 控件 Triggers 元素内的 asp:AsyncPostBackTrigger 元素定义触发器。触发器的控件事件是可选的，如果不指定事件，则触发器事件是控件的默认事件，如对于 Button 控件来说，默认事件是 Click 事件。

为 UpdatePanel 控件指定触发器的示例代码如下：

```
<asp:Button ID="Button1"  Text="Refresh Panel"  runat="server" />
<asp:ScriptManager ID="ScriptManager   runat="server" />
<asp:UpdatePanel ID="UpdatePanel1"
              UpdateMode="Conditional"
              runat="server">
```

```
                <Triggers>
                  <asp:AsyncPostBackTrigger ControlID="Button1" />
                </Triggers>
                <ContentTemplate>
                <fieldset>
                <legend>UpdatePanel content</legend>
                <%=DateTime.Now.ToString() %>
                </fieldset>
                </ContentTemplate>
    </asp:UpdatePanel>
```

在上面的示例中，单击 Button1 控件后，UpdatePanel1 内的内容就会刷新。

3）Timer 控件

ASP.NET Ajax Timer 控件可按照定义的间隔执行回发。如果将 Timer 控件和 UpdatePanel 控件结合在一起使用，可以按照定义的间隔启用部分页更新。使用 Timer 控件也可以发布整个网页。

Timer 控件的常用属性如下：

✧ Interval：间隔时间，单位为毫秒，每一个间隔时间后将触发 Tick 事件。

注意：Timer 要放在其所刷新的 UpdatePanel 内部，放外面的话要设置 UpdatePanel 的 Triggers 属性。

四、任务拓展

本节完成一个课外拓展实践任务。

拓展任务卡 2

拓展任务号	6-2	任务名称	后台管理的其他功能实现
计划用时	120 分钟	任务性质	课外
任务描述与目标			
实现聊天室后台部分的会员注册、会员管理、管理员退出等功能；学习 Ajax 服务端控件的使用			
主要操作步骤提示			
1．设计会员注册页，编写注册事件代码； 2．设计会员管理页，在该页中使用 ScriptManager 及 UpdatePanel 控件支持页面的局部更新操作，编写后台事件代码； 3．测试			

6.3 任务 2 ASP.NET Ajax 服务器控件应用

任务描述

本任务主要是通过实现聊天室的前台聊天业务，涉及了注册会员、会员登录、选择聊天室、发送聊天消息、浏览在线聊天用户、定时刷新聊天消息等功能的实现进一步学习 ASP.NET Ajax 服务器控件的应用。

解决方案

为完成本任务，要完成以下几个方面的工作：

（1）能够在 Web 用户控件中添加自定义事件及属性；

（2）能够使用 ASP.NET Ajax 服务端控件实现页面信息的局部更新。

6.3.1 会员注册

一、实战演练

（1）在项目中添加全局应用程序类文件 Global.asax。

（2）在 Global 类中定义如下两个成员变量，分别用于存储系统中所有开放状态的聊天室和每个聊天室的在线会员信息，具体代码如下：

```
        public List<ChatRoomInfo> roomList;                    //保存开放的聊天室列表
        public Dictionary<int, List<UserInfo>> userList;       //记录每个聊天室内的在线会员信息
```

（3）编写 Application_Start 事件，实现聊天室的初始化，具体代码如下：

```
        ChatRoomDAL chatDAL = new ChatRoomDAL();
        roomList=chatDAL.GetAllOpenRoom();
        userList = new Dictionary<int, List<UserInfo>>();
        foreach (ChatRoomInfo room in roomList)
        {
            userList.Add(room.ChatRoom_ID, new List<UserInfo>());
        }
        //保存开放的聊天室列表
        Application["userList"] = userList;
        //保存每个聊天室内的在线会员信息
        Application["roomList"] = roomList
```

（4）在项目根目录下创建会员注册页面 Register.aspx。设计效果如图 6-8 所示。

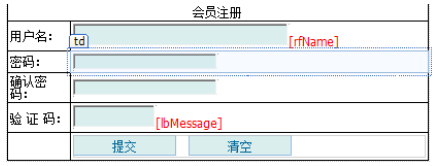

图 6-8　会员注册页面设计效果

（5）编写用户注册事件代码，具体代码如代码清单 6-5 所示。

代码清单 6-5　用户注册代码

```
//“提交”按钮的 Click 事件响应代码
protected void btnRegister_Click(object sender,EventArgs e)
    {
      if(Session[ValidateCode.VALIDATECODEKEY] != null)
        {   //验证验证码是否正确
            if(tbCode.Text != Session[ValidateCode.VALIDATECODEKEY].ToString())
            {
                lbMessage.Text = "验证码输入错误，请重新输入";
                return;
            }
//注册用户，注册成功则转入聊天室登录页面，如果注册失败，给出注册失败信息
UserInfoDAL userInfoDAL=new UserInfoDAL();
//调用业务类 UserInfoDAL 中的 AddNewUser 方法，实现注册功能
```

```
    if (userInfoDAL.AddNewUser(tbUsername,tbPassword))
                {//注册成功
              ScriptManager.RegisterClientScriptBlock(this,    this.GetType(),
String.Empty, "alert('会员注册成功！')", true);
              Response.Redirect("~/Login.aspx");
            }
    else
         {
     ScriptManager.RegisterClientScriptBlock(this,
     this.GetType(), String.Empty, "alert('会员注册失败，请重新注册！')", true);
            return;
    }
    }
```

（6）测试运行。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-8 所示。

表 6-8　演练完成情况评价表

任务号	6-5	任务名称	会员注册
任务子项	完成情况	主要问题	未完成原因
设计全局应用程序类 Global.asax			
设计会员注册页，编写事件代码			
测试			

6.3.2　选择聊天室登录

聊天室只对注册会员开放，并且聊天室里的聊天用户不能超过最大人数。选择聊天室登录涉及了会员的登录验证及是否可进入指定聊天室的判定。

一、实战演练

（1）打开项目中的 Default.aspx 页，设计会员登录界面，设计效果如图 6-9 所示。

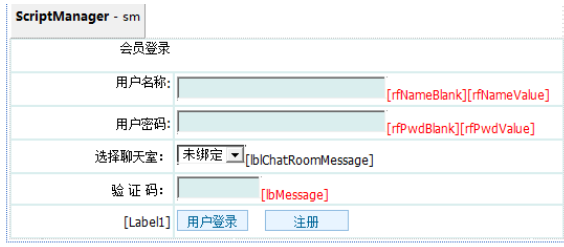

图 6-9　会员登录界面设计效果

（2）编写用户登录事件代码，如代码清单 6-6 所示。

代码清单 6-6　会员登录事件代码

```
    protected void btnLogin_Click(object sender,EventArgs e)
    {
         //此处省略了验证码验证部分的代码，具体可参见注册事件代码
           ChatRoomInfo room = FindRoomById(int.Parse(drpRoomList.SelectedItem.
Value));
           if (room.ChatRoom_CurrentNum >= room.ChatRoom_MaxNum)
           {
             //弹出“房间已满员，禁止登录!”的信息
              ScriptManager.RegisterClientScriptBlock(this,
              this.GetType(), String.Empty, "alert('房间已满员，禁止登录！')",
true);
              return;
           }
           //可进入聊天室，则需修改两项内容
           UserInfoDAL userDAL = new UserInfoDAL();
           int uid = userDAL.GetAUser(tbUsername.Text, tbPassword.Text);
    if (uid > 0)
           {//用户验证成功
              UserInfo user0 = new UserInfo();
              user0.ChatRoom_ID = room.ChatRoom_ID;
              user0.User_ID = uid;
              user0.User_Name = tbUsername.Text;
           //保存信息
              FormsAuthentication.SetAuthCookie(tbUsername.Text, false);
              Session["id"] = user0;
              Session["chatroom"] = room.ChatRoom_Name;
              Application.Lock();
           //修改登录房间的在线人数
              if (lists != null && lists.Count > 0)
           {
              foreach (ChatRoomInfo room1 in lists)
              {
                 if (room1.ChatRoom_ID ==room.ChatRoom_ID)
                 {
                    room1.ChatRoom_CurrentNum++; break;
                 }
              }
           }
           //获取指定房间的用户列表，加入用户
              Dictionary<int, List<UserInfo>> userList = (Dictionary<int,
    List<UserInfo>>)Application["userList"];
                        List<UserInfo> users;
               bool ret = userList.TryGetValue(room.ChatRoom_ID, out users);
               if (ret)
               {
                   UserInfo user1=new UserInfo();
                   user1.ChatRoom_ID=room.ChatRoom_ID;
                   user1.User_ID=uid;
                   user1.User_Name=tbUsername.Text;
                   users.Add(user1);
               }
              //保存到 Application 中
               Application["userList"] = userList;
               Application["roomList"] = lists;
               Application.UnLock();
              Response.Redirect("~/Member/Chat.aspx");
           }
       }
```

（3）测试运行。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-9 所示。

表 6-9 演练完成情况评价表

任务号	6-6	任务名称	选择聊天室登录
任务子项	完成情况	主要问题	未完成原因
登录界面设计			
事件代码编写			
测试			

6.3.3 即时显示在线人员信息

一、实战演练

（1）在项目的 Member 目录下添加名为 Chat.aspx 的 Web 窗口，该窗口作为聊天的主界面。该页面的整体结构如图 6-10 所示。

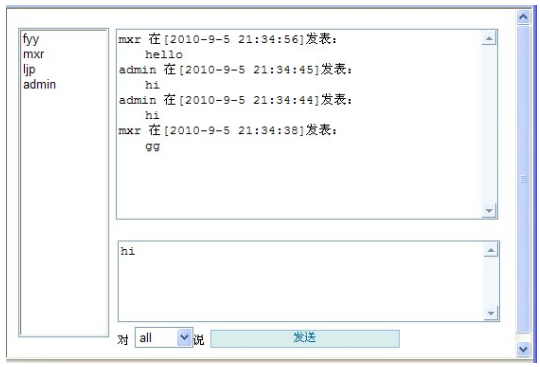

图 6-10 聊天主界面的整体结构

（2）设计聊天室在线会员区部分页面，在线会员区的 HTML 代码如代码清单 6-7 所示。

代码清单 6-7 在线会员区的 HTML 代码

```
    <asp:LinkButton ID="lbtnLogout" runat="server" onclick=" lbtnLogout _Click">
注销</asp:LinkButton>
       <asp:UpdatePanel ID="UpdatePanel1" runat="server">
          <ContentTemplate>
    <asp:Repeater ID="Repeater1" runat="server"
                    onitemcommand="Repeater1_ItemCommand"  >
    <HeaderTemplate>
    <table>
    </HeaderTemplate>
    <ItemTemplate>
    <tr><td>
    <asp:LinkButton Text='<%# Eval("User_Name") %>'
             CommandArgument='<%# Eval("User_ID")+":"+Eval("User_Name") %>'
             CommandName="select"  runat="server"></asp:LinkButton>
    </td></tr>
```

```
    </ItemTemplate>
    <FooterTemplate></table></FooterTemplate></asp:Repeater>
      <asp:Timer  runat="server"  ID="timer1"  Interval="1000"  ontick="timer1_
Tick"/>
     </ContentTemplate>
    </asp:UpdatePanel>
```

（3）为后台在线会员区添加功能事件代码，如代码清单 6-8 所示。

代码清单 6-8　会员区功能代码

```
      protected void Page_Load(object sender, EventArgs e)
      {
          if (!IsPostBack)
              DataBinds2();
      }
      private void DataBinds2()
      {
          List<UserInfo> users=(List<UserInfo>)Application["online"];
          this.Repeater1.DataSource = users;
          this.Repeater1.DataBind();
      }
   protected void timer1_Tick(object sender, EventArgs e)
      {//定时刷新数据
          if(timer1.Enabled)
          Bind2();
       }
```

（4）测试运行。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-10 所示。

表 6-10　演练完成情况评价表

任务号	6-7	任务名称	即时显示在线人员信息
任务子项	完成情况	主要问题	未完成原因
界面设计			
事件代码编写			
测试			

6.3.4　发送聊天信息

一、实战演练

（1）打开项目中的 chat.aspx 页，设计与发送聊天消息相关的界面，界面设计如图 6-11 所示。

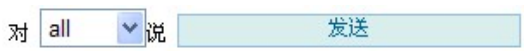

图 6-11　发送聊天消息的界面设计

（2）编写事件代码，实现消息的发送。

① 修改后台代码中的 DataBinds2()方法，实现发送消息区中的接收方用户列表的数据绑定。修改后的代码如下所示，用于接收方用户列表的数据绑定。

```
    private void DataBinds2()
        {
            List<UserInfo> users=(List<UserInfo>)Application["online"];
            this.drpToUsers.DataSource=users;
```

```
        this.drpToUser.DataTextField="User_Name";
        this.drpToUser.DataValueField="User_ID";
this.Repeater1.DataSource = users;
        this.Repeater1.DataBind();
    }
```

② 编写“发送”按钮的 Click 事件代码，实现发送聊天消息的功能，代码如下所示：

```
protected void btnSendMessage_Click(object sender, EventArgs e)
    {
      //发送消息整个过程分为两个步骤，即把消息写入到数据库和把消息添加到聊天区的顶部
      //写入数据库部分通过调用业务类方法实现
      MessageInfoDAL messageDAL=new MessageInfoDAL();
      UserInfo curUser=(UserInfo)Session["id"];       //获取当前用户
      messageDAL.SendMessage(curUser.User_ID, drpToUsers.SelectItem.Value,
                        txtMessage,
                        curUser.Chat_ID);
        }
```

（3）运行项目，测试聊天的发送功能。

二、任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-11 所示。

表 6-11 演练完成情况评价表

任务号	6-8	任务名称	发送聊天消息
任务子项	完成情况	主要问题	未完成原因
设计发送聊天消息界面			
事件代码编写			
测试			

6.3.5 聊天消息的定时刷新

一、实战演练

（1）打开项目中的 chat.aspx 页，设计与显示聊天消息相关的界面，界面设计如图 6-12 所示。

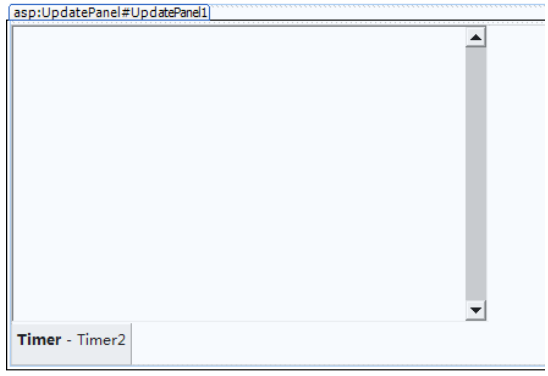

图 6-12 显示聊天消息的界面设计

（2）该部分的 HTML 源代码如下所示：

```
    <asp:UpdatePanel ID="UpdatePanel1" runat="server">
          <ContentTemplate>
       <asp:TextBox ID="txtMessages" runat="server" Height="247px" TextMode="
MultiLine"
                  Width="391px"></asp:TextBox>
    <asp:Timer runat="server" ID="Timer2" Interval="1000">
    <!—这里通过设置 Interval 属性实现了每秒刷新一次操作 -->
          </asp:Timer>
          </ContentTemplate>
         </asp:UpdatePanel>
```

（3）添加与定时刷新聊天消息功能相关的事件代码，具体代码如代码清单 6-9 所示。

代码清单 6-9　定时刷新聊天消息功能代码

```
    protected void timer1_Tick(object sender, EventArgs e)
        {   //定时刷新数据
           if(timer2.Enabled)
           {//调用业务类，实现当前聊天室中的最新 25 条记录
            MessageInfoDAL messageDAL=new MessageInfoDAL();
             DataSet ds= messageDAL. GetRecentMessage((UserInfo)Session["id"].
Chat_ID);
           //把记录添加到消息显示区
           (Message_PubTime)[uFrom]对[uTo]说：MessageBody
           //显示消息的格式为(发表时间)[发送方]对[接收方]说：消息内容
           DataTable table=ds.Tables[0];
    txtMessages.Text="""";       //清除原有的聊天消息
           for(int i=0;i<table.Rows.Count;i++)
              {
               string pubtime=(string)table.Rows[i]. ItemArray[2];   //发表时间
              string uFrom=(string)table.Rows[i]. ItemArray[3];      //发送方
    string uTo=(string)table.Rows[i]. ItemArray[4];                  //接收方
    string messageBody=(string)table.Rows[i]. ItemArray[1];          //消息
             txtMessages.Text=  txtMessages.Text  +string.Format("({0})[{1}] 对
[{2}]说:{3} \n",
                                      pubtime,uFrom,uTo,messageBody);
      }}}
```

（4）测试运行。

任务完成情况评价

学生在老师的演示和指导下，对完成情况进行自评，情况评价表如表 6-12 所示。

表 6-12　演练完成情况评价表

任务号	6-9	任务名称	聊天消息的定时刷新
任务子项	完成情况	主要问题	未完成原因
设计聊天消息显示界面			
事件代码编写			
测试			

拓展任务跟踪卡

任务号		任务名称	
合作人员			
开始时间	结束时间	计划时间	实际时间
完成情况描述			

项目完成评价

项目号			项目名	
项目完成方式			□小组协作 □个人独立	
项目完成情况（60%）	界面设计（5%）	自我评价		
		小组评价		
		个人评价		
	代码编写（20%）	自我评价		
		小组评价		
		个人评价		
	功能实现（30%）	自我评价		
		小组评价		
		个人评价		
	数据库设计（5%）	自我评价		
		小组评价		
		个人评价		
拓展与创新（40%）	创新能力（20%）	自我评价		
		小组评价		
		个人评价		
	设计潜力（20%）	自我评价		
		小组评价		
		个人评价		
主要存在问题				

课外思考题

1．Ajax 技术的关键技术有哪些？Ajax 有何优缺点?

2．ASP.NET 提供了哪些 Ajax 服务端控件？这些控件各自有什么作用？

3．利用 ASP.NET 提供的 Ajax 服务端控件实现一个基于 Ajax 的网上投票系统。